Rechnergestützte Übergänge zwischen gegenständlichen und abstrakten Modellen produktionstechnischer Systeme: erster und zweiter Zwischenbericht zum DFG Forschungsprojekt RUGAMS

Schäfer, Kai; Bruns, Friedrich Wilhelm; Brauer, Volker

Veröffentlichungsversion / Published Version
Zwischenbericht / interim report

Zur Verfügung gestellt in Kooperation mit / provided in cooperation with:
SSG Sozialwissenschaften, USB Köln

V. Brauer, W. Bruns, K. Schäfer

Rechnergestützte Übergänge zwischen gegenständlichen und abstrakten Modellen produktionstechnischer Systeme

Erster und zweiter Zwischenbericht zum DFG Forschungsprojekt RUGAMS

artec Arbeitspapier 56, Februar 1998

Erster Zwischenbericht, Zeitraum: April bis Dezember 1996
Zweiter Zwischenbericht, Zeitraum: Januar bis Dezember 1997

Erster Zwischenbericht zum DFG Forschungsprojekt **RUGAMS**

Prof. Dr.-Ing. F. Wilhelm Bruns
Universität Bremen
Forschungszentrum Arbeit und Technik (artec)
Dezember 1996

Forschungsvorhaben:
Rechnergestützte Übergänge zwischen gegenständlichen und abstrakten Modellen produktionstechnischer Systeme (RUGAMS)

DFG Gz.: BR 1556/2-1

Berichtszeitraum:
April bis Dezember 1996

Projektgruppe:
Prof. Dr.-Ing. F. Wilhelm Bruns
Dipl.-Inform. Volker Brauer
Dipl.-Ing. Kai Schäfer
Cand.-Inform. Martin Faust

Universität Bremen
Forschungszentrum Arbeit und Technik (artec)
Bibliothekstr. (MZH)
28359 Bremen

Tel.: 0421-218-4206 (-2435, Sekretariat)
Fax: 0421-218-4449
email: bruns@artec.uni-bremen.de

Inhalt

1. Problemlage und Motivation

Die rechnergestützte Simulation, als zielgerichtetes Modellieren und Experimentieren, ist ein Verfahren, das bereits erkenntnisfördernd in der Auslegungsplanung komplexer Logistik-, Materialfluß- und Produktionssysteme eingesetzt wird. Zahlreiche Arbeitskreise, Tagungen und Veröffentlichungen belegen die vielfältigen Anwendungen und Einsatzmöglichkeiten der Simulationstechnik (vgl. ASIM, EUROSIM und SCSI[1], Kuhn & Reinhardt 1993). Noche & Wenzel (1991) weisen in einem Marktspiegel 74 verschiedene Simulatoren für den Bereich Produktion und Logistik aus.

Der Ursprung dieser Systeme liegt in Entwicklungen wie GPSS (vgl. Gordon 1981), SIMULA (vgl. Nygaard 1981) und SMALLTALK (vgl. Goldberg 1983), deren Konzepte inzwischen auch wirkungsvollen Eingang in universelle Programmiersprachen gefunden haben. Obwohl die Vielfalt und Mächtigkeit der Simulatoren beständig zunimmt, siehe Simple++ (vgl. Becker 1991), USE! (vgl. Wenzel 1993), DEVS (vgl. Zeigler 1990), ist die Marktdurchdringung der Simulation nicht sehr hoch. Als Grund wird häufig der hohe Aufwand von Simulationsstudien oder die ungenügende Überprüfbarkeit der Modellvalidität angegeben (vgl. Splanemann 1995).

In einer Befragung zu Schwachstellen der Simulationstechnik bestätigen Anwender folgende Aussagen (Splanemann 1995, S. 14):

1. Es fehlt eine ausreichende Integration der Simulationstechnik in die betrieblichen Abläufe.
2. Die Einsatzmöglichkeit der Simulation für die gegebene Aufgabenstellung ist nicht bekannt.
3. Die Simulation ist zu aufwendig bezüglich der Datenbeschaffung.
4. Die Ausbildung im Bereich der Simulationstechnik ist unzureichend.
5. Die Simulationssystemvertreiber/-anbieter machen zu positive Versprechungen bzgl. der Leistungsfähigkeit und Leichtigkeit des Simulationseinsatzes.
6. Die Simulation ist zu aufwendig bezüglich des Personaleinsatzes bzw. Zeitbedarfs.
7. Simulation wird als Spielerei angesehen, als wenig glaubwürdig.
8. Am Markt verfügbare Simulationssysteme wirken aufgrund ihrer hohen Komplexität abschreckend.
9. Der Widerspruch zwischen Mächtigkeit bzw. Flexibilität und Bedienungskomfort von Simulationssystemen ist bisher nicht zufriedenstellend gelöst.

Einerseits werden also Informations- und Qualifikationsdefizite auf Seiten der Anwender festgestellt, andererseits erfordert die Simulationstechnik einen als zu hoch eingeschätzten betrieblichen Aufwand bei geringer Glaubwürdigkeit. Eine Weiterentwicklung simulationstechnischer Methoden und Werkzeuge mit dem Ziel der Verminderung des Aufwandes und der Erhöhung ihrer Glaubwürdigkeit liegt nahe.

Bei einer Zerlegung des Simulationsprozesses in die Phasen:

1. Problemformulierung und Zielsetzung
2. Strukturanalyse und Alternativenbildung
3. Datenerhebung und Datenaufbereitung
4. Modellerstellung
5. Modellüberprüfung und Validation

[1]Arbeitsgemeinschaft Simulation in der Gesellschaft für Informatik,
Federation of European Simulation Societies,
The Society for Computer Simulation International

6. Simulationsläufe und Analyse
7. Bewertung und Ergebnispräsentation

kommen Scharf (1990) und Zühlke (1994) auf der Basis von Anwenderbefragungen zu *relativen Aufwänden* von 51% bzw 57% für die Modellierung (Phasen 3-5) (Splanemann 1995).

Zur *Glaubwürdigkeit:* Bei der Lösung eines vorgegebenen einfachen Referenzproblems (Flexible Assembly System in Eurosim) durch unterschiedliche Anwender mit unterschiedlichen Simulationssystemen ergaben sich z. T. erheblich differierende Ergebnisse (vgl. Krauth 1993). Diese Differenzen zeigen u.a., wie abhängig Simulationen von unterschiedlichen Detailinterpretationen der Aufgabenstellung sein können. Es ist bekannt, daß bei Systemen mit nebenläufigen Prozessen das Gesamtverhalten stark von den lokalen, zunächst marginal erscheinenden Synchronisations-, Konfliktlösungsregeln und Vereinfachungen abhängen kann und Ergebnisse entstehen, die weit von der Realität entfernt liegen. Fuss (1994, S. 96) charakterisiert diese Tatsache zugespitzt: "Halbgeordnete Probleme löst man am bequemsten, indem man Probleme hinausdefiniert - und damit eine zwar ähnliche, aber andere, i.a. leichtere Aufgabe löst". Diese Erkenntnis spricht jedoch nicht gegen das erhellende Mittel der Simulation insgesamt, sie hebt lediglich die besondere Bedeutung einer sorgfältigen Systemspezifikation und Modellierung, sowie einer reflektierten Interpretation der Ergebnisse hervor.

Verschiedene Arbeiten haben die Reduktion des Modellierungsaufwandes und der Fehlermöglichkeiten im engeren (datentechnischen) Sinne zum Ziel. Klußmann (1994) und Splanemann (1995) zeigen für einen eingeschränkten Anwendungsbereich prototypisch die Übernahme bereits existierender betrieblicher Daten aus anderen DV-Systemen wie CAD und PPS über ein systemneutrales Austauschformat und deren Nutzung für die teilautomatische Generierung von Simulationsmodellen. Andere Konzepte versuchen den Modellierungsteil dadurch zu effektivieren, daß flexible, modulare Bausteinkonzepte entwickelt werden, die anwendungsspezifisch konfigurierbar sind (Simple++) und hierarchische, selbstenthaltende (endomorphe) Strukturen abbilden können (vgl. Zeigler 1990).

Weitergehende Ansätze entstammen der "Künstlichen Intelligenz" - Forschung und der Raumfahrttechnik. Am Knowledge Systems Laboratory (Stanford University) wird im "How Things Work"-Projekt (vgl. Fikes, et al 1994) an einer "intelligenten" Gerätemodellierungsumgebung gearbeitet. Diese soll die automatische Modellgenerierung, Modellverifikation und Generierung umgangssprachlicher Ergebniserklärungen aus formalen Anforderungs-, Funktions- und Verhaltensspezifikationen ermöglichen (vgl. Kuipers 1994, S. 321-380, Vescovi et al 1993, Iwasaki & Chandrasekaran 1992, Gruber & Gautier 1993). Die im Zusammenhang mit dem amerikanischen Verbundforschungsprojekt MADE (Manufacturing Automation and Design Engineering) zu sehenden Forschungen befinden sich noch in einem frühen Stadium, trotzdem sind einzelne praxisrelevante Ergebnisse bereits sichtbar (vgl. Finin et al 1993). In diesen Forschungen ist das Ziel erkennbar, komplexe technische Systeme mit einem integrierten Modellierungsinstrumentarium zu beschreiben, das kontinuierliche und diskrete Sichtweisen unterstützt und die bisher wenig in der produktionstechnischen Praxis eingesetzte *Qualitative Simulation* (vgl. Fishwick & Luker 1991, Kuipers 1994) einbezieht. Zwar werden die formalen Methoden im Sinne ihrer Anwendungsbreite (Problemräume) und sequentiellen Überdeckung der Simulationsphasen (Spezifikation bis Ergebnisinterpretation) erweitert, es ist jedoch fraglich, ob hierdurch der intuitive, iterative, kooperative, die Einsicht fördernde Simulationsprozeß unterstützt oder behindert wird.

Eine Alternative zur zunehmenden Formalisierung und Automatisierung des Modellierungsprozesses, die sowohl den Modellierungsaufwand reduzieren als auch die Glaubwürdigkeit erhöhen kann, sehen wir in der Entwicklung praxisrelevanter Konzepte, die einen schrittweisen Übergang von natürlichsprachlicher, vager Beschreibung der Problem- und Zielsituation in Verbindung mit konzeptuellen Modellen über formale funktionale und strukturelle Modelle bis zum rechnerinternen lauffähigen Modul unterstützen.

Zwei Ansätze, den Modellierungsvorgang anschaulicher, konkreter, handhabbarer und diskursiver - auch im Sinne einer schnellen Prototypentwicklung qualitativ unterschiedlicher Entwurfsvarianten - zu gestalten, können unterschieden werden.

1. Es werden Modellelemente im Rechner angeboten, die in ihrer dreidimensionalen Form und ihren räumlichen Beziehungen zueinander möglichst große Ähnlichkeit zu den realen Objekten haben. Über Interfacetechniken aus dem Bereich der "Virtuellen Realität", wie Datenhandschuh und Datenbrille (vgl. Warnecke 1994) werden diese Modellelemente so manipuliert, als befände sich der Modellierende zwischen und in ihnen (imersive computing) (vgl. Isdale 1993). Eine Variante dieses Ansatzes ist es, die virtuellen Modellelemente in den realen Raum zu projezieren und schattenspielartig mit den Händen zu verändern (vgl. Krueger 1991).

2. Es wird eine direkte Kopplung zwischen abstrakten, im Rechner abgebildeten Modellen und konkreten physikalischen Modellen angestrebt. Reinhardt (1988) realisierte die Möglichkeit, ein rechnerinternes SPS-Steuerungsmodell schrittweise in ein physikalisches Modell zu exportieren. Insbesondere bei der Layoutplanung neuer technischer Anlagen, wie zur Schiffsmotorfertigung, zur Expressgutverteilung, zum Transport von Containern und deren Verladung in Terminals hat sich die konkrete Repräsentation der Anlagen in Form gegenständlicher Modelle als förderlich für den Verständigungsprozeß in interdisziplinär zusammengesetzten Gestaltungsteams erwiesen (vgl. Scheel et al 1995).

An einer synchronen Kopplung zwischen der Modellierung im Realen und der rechnerinternen Modellgenerierung arbeiten verschiedene Forschungsgruppen. Kang & Ickeuchi (1994) entwickeln ein Konzept des Roboterprogrammierens durch Vormachen im Realen. Über die Aufzeichnung und Erkennung von Handposen und -gesten wird ein Roboterprogramm zur Montagearbeit generiert. Bruns et al (1994) versuchen in einem allgemeineren Ansatz, diese Technik für den Verständigungsprozeß bei der Modellierung von Materialfluß- und Produktionssystemen einzusetzen und mit existierenden Simulatoren in einem koppelbaren Instrumentarium zu verbinden.

2. Zielsetzung

Ziel des Projektes ist die Entwicklung eines rechnergestützten Modellierers, der es ermöglicht, mit stofflichen Bausteinen ein gegenständliches Modell eines technischen Systems aufzubauen und synchron dazu ein strukturell und funktional repräsentatives Modell im Rechner zu erzeugen, das dort leicht variierbar und analysierbar ist. Über einen Datenhandschuh erfaßte Hand- und Fingerbewegungen führen zu Operationen auf bereits vorhandenen rechnerinternen Modellfragmenten, die zu einem Gesamtmodell zusammengesetzt und in Simulationsexperimenten weiter verwendet werden können. Damit soll der Modellierungsprozeß um eine Dimension der Konkretheit erweitert werden, die für das Verständnis komplexer räumlicher und funktionaler Zusammenhänge hilfreich ist.

Die Ausgangsthese des Projektes ist, daß der Grad der Stofflichkeit und Konkretheit von Modellierungselementen einen Einfluß auf die Herausbildung geeigneter mentaler Modelle bei den Modellierenden hat. Dies wirkt sich auch auf die Kommunikation unter ihnen und auf die Überprüfung der Modellgültigkeit sowie der Modellangemessenheit aus. Verständnis, Verständigung und Kritik sind wichtige Dimensionen der Technikgestaltung. In verschiedenen Simulationsprojekten stellten wir auf Seiten der Anwender eine überzogene Technikgläubigkeit oder aber -skepsis fest, die mit der Tatsache verbunden war, daß der Bezug zwischen rechnerinternen, abstrakten, nichtstofflichen Zusammenhängen und der konkreten Realität nicht ausreichend transparent und überprüfbar war. Ziel dieses Projektes ist daher die Entwicklung von *Modellübergängen*, die eine schrittweise, leichter

verifizierbare und validierbare Modellierung unterstützen und zugleich den Modellierungsvorgang kommunikativer gestalten, weil er im Realen und nicht vor dem und über den Rechner stattfindet.

Aufbauend auf den im Forschungszentrum artec vorliegenden konzeptionellen Arbeiten zu einer Werkstattmetapher und den realisierten Softwarekomponenten zur Gestenerkennung, Geometriedatenverarbeitung und diskreten ereignisorientierten Simulation ist die prototypische Realisierung eines stofflichen und virtuellen Simulationsbaukastens (Labilbaukasten LABILO) geplant. Ein Labilbaukasten ist ein physischer Baukasten mit erweiterbaren Maschinenelementen und jeweils entsprechenden Modellfragmenten im Rechner. Unter Verwendung eines Datenhandschuhs wird das Zusammenfügen der mechanischen Elemente von einem Prozeß der Modellbildung im Rechner begleitet, der aber weitgehend im Rücken der Modellierenden stattfindet. Durch Gesten und Griffe wird der rechnerinterne Modellbildungsprozeß gesteuert. In der modellerzeugenden Phase besteht eine direkte geometrische, topologische und funktionale Analogie zwischen dem physikalischen und dem rechnerinternen Modell. Erst in der nachfolgenden Phase des Experimentierens mit dem rechnerinternen Modell und dessen Variation kann eine Auseinanderbewegung der beiden Modelle erfolgen. Das Projekt will zunächst die grundsätzliche Machbarkeit des Konzeptes an einem funktionsfähigen Prototypen demonstrieren. Dieser Prototyp kann einfache Transportprozesse abbilden. Damit sollen Erfahrungen gewonnen werden, die für die Behandlung komplexerer Modelle, wie der Montage, der Steuerungstechnik, der Maschinenbenutzungsoberfläche u.a. hilfreich sind.

3. Arbeitsplan

Auf dem Weg zum Projektziel, eine Modellierumgebung zu realisieren, waren zunächst u.a. grundlegende technische Probleme zu lösen. Diese bezogen sich insbesondere auf die Inbetriebnahme und Integration spezieller Hard- und Software-Komponenten. Folgende Arbeitspakete wurden behandelt und zum Teil bereits realisiert:

1. Auswahl eines geeigneten Datenhandschuhs. Zwei existierende Handschuhtypen (5th Dimension und TUB-Sensorhandschuh) wurden auf ihre Eignung und Integrationsmöglichkeit untersucht werden.
2. Anpassung eines neuen Datenhandschuhs an die PC-Hardware und Integration in die vorhandene Modellierungssoftware.
3. Weiterentwicklung der Grifferkennung. Im geplanten Vorhaben wurde die Grifferkennung verallgemeinert, indem das von Brauer (1994) entwickelte teachbare, feature-orientierte Klassenkonzept integriert wurde. Hierbei handelt es sich um die Erzeugung eines Merkmalraumes, der durch mehrmaliges Vormachen eines Griffes gebildet wird und im späteren Erkennungsprozeß auf der Basis einer Varianzanalyse zur Identifizierung eines unscharfen Griffspektrums dient.
4. Erweiterung des Drahtgittermodellierers auf einen Berandungsflächenmodellierer.
5. Erweiterung der geometrischen Modellbausteineigenschaften um dynamische, die relevant für ein Simulationsanwendungsgebiet wie Transport und Logistik sind (Quelle, Senke, Bewegung usw.).
6. Kombination des geometrischen Modellierers mit Simulationskernfunktionen. Eine weitere Funktionserweiterung des Modellierers wurde durch die Anbindung der Simulatoren COSIMIR und GHOST erreicht.
7. Konzeptentwicklung für eine akustische Rückkopplung.
8. Entwicklung der Bausteine für den Prototypen (Förderbandelemente, Werkstückträger). Da die Funktionalität des Prototypen an dem Referenzmodell eines Flexiblen Montagesystems nach EuroSim demonstriert werden soll, sind Modellelemente für Geraden und winkelförmigen Trans-

port, Weichen, Quellen, Senken, Bearbeitungsstationen, Werkstücke und Werkstückträger virtuell und real zu modellieren.

4. Vorläufige Ergebnisse

Während des Berichtszeitraumes entstand ein erster Prototyp eines gegenständlichen Modellierers (Abb.1). Dieser demonstriert bereits einige wesentliche Konzepte des Projektes.

Der Prototyp besteht aus einem gegenständlichen Fischertechnik-Modell einer Förderbandanlage, einem PC, in dem das Modell virtuell gespeichert wird und grafisch dargestellt werden kann, und einem Datenhandschuh. Das Konzept des gegenständlichen Modellierens besteht in diesem Zusammenhang darin, Operationen wie z.B. den Transport von Paletten am Fischertechnik-Modell manuell vorzumachen, die Aktionen der Benutzerhand mit dem Datenhandschuh zu erfassen und das rechnerinterne Modell zu aktualisieren.

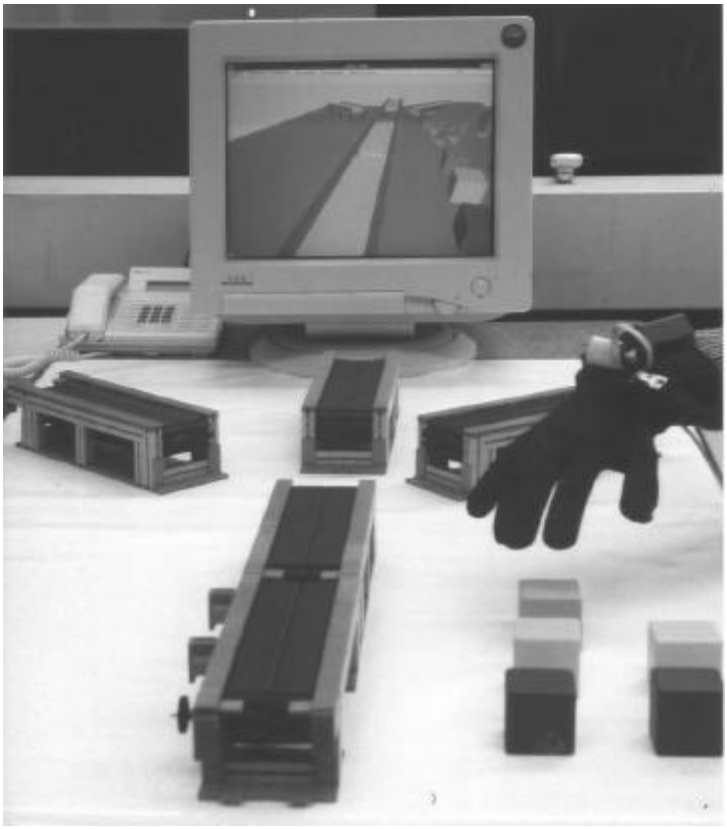

Abbildung 1: Eine prototypische Transportanwendung

Der gesamte Prozeß des Modellierens wird mit Hilfe unserer Software analysiert und weiterverarbeitet. Wir können zeigen, daß durch einfaches Vormachen a) die Regeln für die Verteilung unterschiedlicher Paletten durch ein Förderbandsystem erzeugt werden können und b) ein Roboterprogramm für das Umsetzen zwischen räumlich getrennten Förderbändern generiert werden kann. Diese Anwendung wurde auf der Industriemesse in Hannover ausgestellt.

Nachfolgend wird der Fortschritt der in Kapitel 3 aufgeführten Arbeitsschritte beschrieben und dokumentiert, welche konkreten Erkenntnisse dabei gewonnen werden konnten. Zudem wird dargelegt, welche Probleme dabei zu bewältigen waren bzw. z.Zt. noch offen sind.

4.1 Auswahl eines geeigneten Datenhandschuhs

Unter den kommerziell verfügbaren Datenhandschuhen stellt der 5th Glove von 5th Dimension Technologies (5DT) einen Kompromiß zwischen vertretbarem finanziellem Aufwand und der gebotenen Funktionalität dar. Low-Cost Datenhandschuhe, wie der PowerGlove von Mattel, genügen professionellen Ansprüchen nicht, wogegen der CyberGlove von Virtex unter den Datenhandschuhen zu den Spitzenprodukten zählt, aber mit ca. 15 TDM zu teuer ist. Der an der TU Berlin (TUB) entwickelte Sensorglove wurde als eine mögliche Alternative zu einem kommerziellen Produkt während eines Besuches an der TUB besichtigt. Die technischen Daten des Gerätes sind in (Hommel et al, 1994) veröffentlicht worden.

Die Funktionalität eines Datenhandschuhs läßt sich grob in zwei Bereiche einteilen. Das sog. Trakking System mißt die Position und die Orientierung eines Objektes (z.B. der Hand) im Raum, und die in das Handschuhmaterial eingearbeitete Sensorik erfaßt die Konfiguration der Hand, d.h. unter anderem die Beugung der Fingergelenke. Sowohl für das Tracking als auch für die Konfigurationsmessung werden unterschiedliche Technologien eingesetzt. Der TUB Sensorglove ist mit einem Ultraschall-Ortungssystem und mechanischen Beugungssensoren, die im wesentlichen aus elektronischen Schiebewiderständen bestehen, ausgestattet. Zudem besitzt er Drucksensoren, mit denen die Berührung der Fingerspitzen mit physikalischen Objekten oder der Handinnenfläche registriert werden kann. Der 5DT Handschuh wird ohne eigenes Tracking System angeboten, kann jedoch problemlos mit einem der marktüblichen Produkte ausgestattet werden. Dafür wurde das Polhemus 3Space IsoTrak II System ausgewählt, welches auf elektromagnetischer Basis funktioniert (vgl. Sturmann & Zeltzer, 1994). Die Beugung der Finger wird mit in den Handschuh eingearbeiteten Lichtwellenleitern gemessen, die in Abhängigkeit von der Fingerkrümmung ihre Lichtdurchlässigkeit ändern. Um die beiden Datenhandschuhe miteinander zu vergleichen, können eine Reihe von technischen Kriterien (siehe Tabelle 1) genannt werden (vgl. auch Sturmann, 1992).

Eigenschaft	TUB Sensorglove	5DT Glove/Polhemus
Anzahl Freiheitsgrade	Pos./Orient.: 6 Finger: 5 (+ 2 Daumenrot.) Druck: 12	Pos./Orient.: 6 Finger: 5 Druck: 0
Auflösung Fingerwerte	8 Bit	8 Bit
Genauigkeit (Abweichung vom Realen)	Position: ca. 1-2 cm Orientierung: k.A.	Position: ca. 0,25 cm Orientierung: ca. 1°
Übertragungsrate (Datenpakete/sec)	Tracking: 10 Finger: 30	Tracking: bis zu 60 Finger: bis zu 200
Verzögerung (Latency)	k.A.	Tracking: 20 msec
Schnittstelle	RS 232	2 RS232

Tabelle 1: Vergleich TUB Sensorglove mit 5DT Glove/Polhemus Tracking System

Der Vergleich der technischen Daten zeigt die Vorteile des 5DT Handschuhs in Kombination mit dem Polhemus Tracking System gegenüber dem TUB Sensorhandschuh, der lediglich mehr Sensorik für die Fingerbewegungen aufweist. Ausschlaggebend für die Entscheidung für den 5DT Handschuh waren die besseren Werte des Tracking Systems, da die präzise Messung der Ortsveränderung von Objekten im Kontext des Projektes ein wesentlicher Aspekt ist. Weitere Vorteile des kommerziellen Produktes sind die kompakte Verarbeitung, Garantiezeiten, technische Wartung und Unterstützung durch den Hersteller bzw. Händler.

Der praktische Betrieb der Geräte hat gezeigt, daß das Tracking System mit hoher Zuverlässigkeit und den technischen Angaben entsprechend operiert. Es ist allerdings zu beachten, daß möglichst störungsfreie Betriebsbedingungen geschaffen werden. Elektromagnetische Felder reagieren empfindlich auf Feldquellen, die etwa von elektronischen Geräten ausgestrahlt werden. Auch größere

Metallgegenstände, wie Heizkörper und Regale beeinflussen das Feld des Tracking Systems und damit die Genauigkeit des Sensors. Daher wurde eine Umgebung geschaffen, die auf die Betriebsbedingungen des Tracking Systems abgestimmt ist, so wurde ein Modelltisch aus Holzteilen angefertigt und es werden ausschließlich metallfreie Bausteinelemente zum Modellieren verwendet. Der Datenhandschuh arbeitet ebenfalls seiner technischen Spezifikation entsprechend, das erste Exemplar war jedoch bereits nach wenigen Betriebsstunden defekt. Der Austausch gegen ein vom Hersteller überarbeitetes Gerät verlief problemlos.

4.2 Anpassung des Datenhandschuhs an die PC-Hardware und Integration in die Modellierungs-Software

Der Datenhandschuh und das Tracking System sind zwei voneinander unabhängige Geräteeinheiten, die jeweils eine serielle Schnittstelle am PC belegen. Damit die PC-Hardware mit der Peripherie Daten austauschen kann, mußten die Übertragungsprotokolle aufeinander abgestimmt werden, so daß die Geräte jeweils einen kontinuierlichen Datenstrom an den PC liefern.

Als Modellierungs-Software wird das World Tool Kit (WTK) für MS Windows benutzt. Mit dieser C-Programmierbibliothek können 3D-Geometrien dargestellt und animiert werden. Neben den Funktionen zur grafischen Darstellung und Verwaltung von Objekten beinhaltet das WTK auch eine abstrakte Schnittstelle zu Eingabegeräten wie dem Datenhandschuh. Der Anschluß dieser Geräte erforderte die Programmierung eines Treibers, mit dem Koordinatentransformationen, Fehlerbehandlungen, dynamische Speicherverwaltung etc. durchgeführt werden können. Dabei waren viele Details, wie z.B. die unterschiedliche Repräsentation von Objektrotationen, zu berücksichtigen. Diese Arbeiten sind im wesentlichen abgeschlossen, jedoch noch ausführlich zu testen. Gegenwärtig kann der Datenhandschuh in Kombination mit dem WTK benutzt werden, um Objekte (reale und virtuelle) durch Greifen und Loslassen zu positionieren. Es ist geplant, eine von der Funktionalität des WTK unabhängige Schnittstelle zu realisieren, damit ein allgemeiner Zugriff auf die Sensorik zur Verfügung gestellt werden kann.

4.3 Weiterentwicklung der Grifferkennung

Der von Brauer entwickelte Algorithmus zur Erkennung von Posen und Griffen (Brauer 1994, 1996) wurde an die neue Sensorik angepaßt, und dabei die ursprüngliche Implementierung um einige Freiheitsgrade (freie Positionierung und Orientierung der Hand im Raum) erweitert.

Der praktische Umgang mit dem Datenhandschuh und einer Beispielanwendung führte zu der Erkenntnis, daß das Greifen von Objekten gleicher Größe mit einem einfachen und zuverlässigen Verfahren erkannt werden kann. In diesem Spezialfall können die Beugungswerte der Finger summiert und mit einem Schwellwert verglichen werden. Ist die Summe größer als dieser Wert und befindet sich die Hand in unmittelbarer Nähe eines Objektes, so gilt es als gegriffen. Der Benutzer kann bei dieser Art der Grifferkennung sehr variabel greifen, lediglich eine bestimmte Summe von Beugungswerten der einzelnen Finger muß erreicht werden.

Bei einer größeren Vielfalt von Objekten ist eine Unterscheidung verschiedener Griffmuster erforderlich. Dafür wurde ein feature-orientiertes, griffspezifisches Verfahren, das für dynamische Gesten bereits existierte (Brauer 1994), auf die Problematik des Greifens realer Gegenstände übertragen.

4.4 Erweiterung des Drahtgittermodellierers auf einen Berandungsflächenmodellierer

Die bis zum Projektbeginn verwendeten Drahtgittermodelle zur Darstellung von grafischen Objekten eignen sich lediglich zur Verdeutlichung grober struktureller Objekteigenschaften und nur bedingt zur Lokalisierung der Objekte im Raum. Ein hoher optischer Ähnlichkeitsgrad zwischen

realer Welt und virtuellem Modell läßt sich mit Flächendarstellungen erzielen. Unter Verwendung analytischer Beleuchtungsverfahren können grafische Objekte, die als Flächenmodell (Polygonlisten) repräsentiert werden, realitätsnah visualisiert werden. Die meisten grafischen Modelliersysteme wie z.B. AutoCAD erlauben die Speicherung von Modelldaten als Flächenmodell. Mit dem Einsatz der Visualisierungs-Bibliothek World Tool Kit (WTK) können grafische Objekte in Form von Berandungsflächenmodellen importiert und auf einem Bildschirm dargestellt werden. Das WTK unterstützt verschiedene Stufen der Darstellungsgenauigkeit, die mit steigender Realitätsentsprechung einen höheren Berechnungsaufwand erfordern. Die Abstufungen sind Drahtgittermodell, Flat Shading, Gouraud Shading und texturierte Darstellungen (vgl. Foley & van Damme, 1990).

Die realitätsnahe Darstellung dreidimensionaler grafischer Objekte in Echtzeit ist nicht mehr die alleinige Domäne des Supercomputing. Auf den einschlägigen Messen CeBIT in Hannover und VRWorld in Stuttgart wurden leistungsfähige PC-Systeme präsentiert und besichtigt, die auf dem Grafikstandard OpenGL basieren. Mit diesen Systemen können produktionstechnische Systeme grafisch modelliert und animiert werden. Die Kombination dieser Technologie mit dem Real Reality Konzept wirft neue Fragen auf, die innerhalb des Projektes verfolgt werden.

Mit der Entscheidung, die Funktionalität des vorhandenen Drahtgittermodellieres in ein kommerzielles Produkt wie das WTK zu übertragen, konnte ein hoher Aufwand von Programmierarbeiten vermieden werden. Da das WTK für diverse Hardware-Umgebungen verfügbar ist, wurde auf diese Weise auch die Übertragbarkeit von Anwendungen auf andere Systeme erreicht. Ein weiterer Vorteil des WTK ist die Verfügbarkeit von Schnittstellen zu grafischen Modelliersystemen - die meisten praktisch relevanten Dateiformate werden unterstützt.

4.5 Erweiterung der geometrischen Modellbausteineigenschaften um dynamische

Die für den Prototypen verwendeten Modellelemente lassen sich in *aktive* und *passive* sowie *unbewegliche* und *bewegliche* Bausteine einteilen. Eine allgemeine Klassifizierung enthält Tabelle 2.

	Aktiv	**Passiv**
Beweglich	Hand Greifer Fahrzeug Mensch	Palette Container
Unbeweglich	Förderband Roboter	Ablagen Lagerfläche Hindernis

Tabelle 2: Einteilung produktionstechnischer Komponenten mit Beispielen

Während die Eigenschaften *beweglich* und *unbeweglich* weitgehend selbsterklärend sind, bedeutet *aktiv*, daß dieser Baustein sich selbst oder andere Bausteine in ihren Eigenschaften, wie z.B. Lage, Gestalt oder Zusammengehörigkeit zu anderen Bausteinen, beeinflussen kann. *Passive* Bausteine können die Eigenschaften anderer Bausteine nicht ändern.

Passive Modellelemente, wie Paletten oder eine Lagerfläche, konnten bereits zum Zeitpunkt der Antragstellung modelliert werden. Bausteine mit aktiven Eigenschaften wurden neu implementiert. Zusätzlich zur reinen Ortsveränderung von Objekten (Paletten) wird der gesamte Vorgang der Bewegung als Koordinatensequenz (Pfad) gespeichert und einem virtuellen Objekt zugeordnet (siehe Abb. 2). Zudem besitzen die Förderbänder ein dynamisches Grundverhalten, das den Transport von Paletten in eine bestimmte Richtung abbildet.

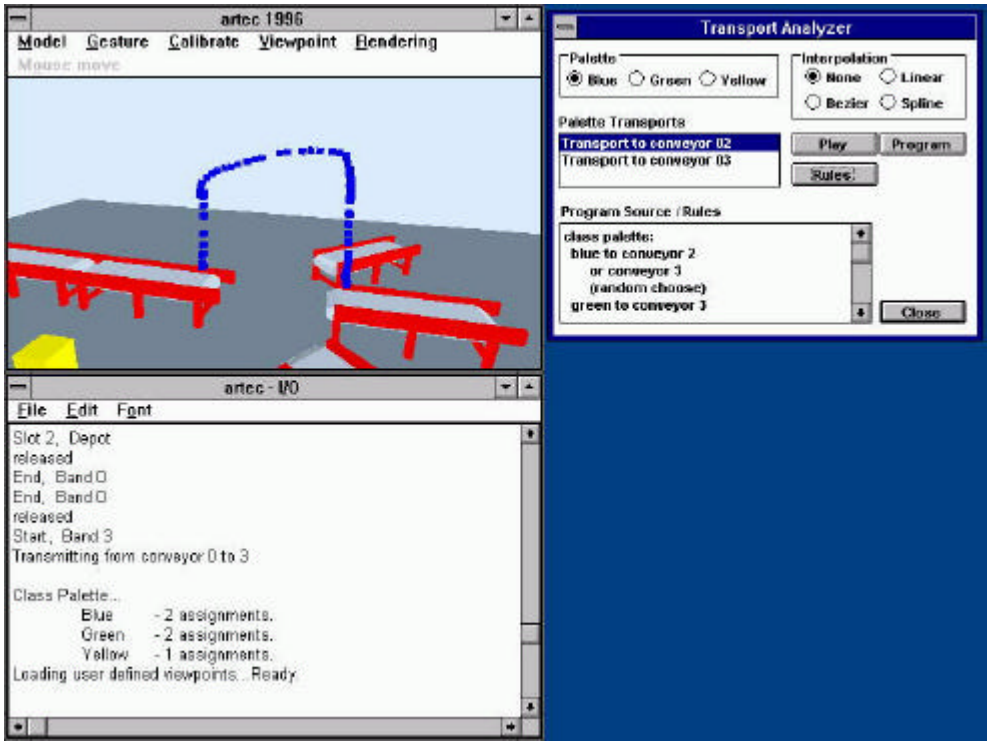

Abbildung 2: Darstellung des Pfades einer virtuellen Palette mit Editorfunktionen

Die aufgezeichneten Pfade dienen als Grundgerüst für ein parametrisches Roboterprogramm. Auf diese Weise simuliert die Hand des Benutzers die Funktion eines Roboters, der die Paletten anhand eines bestimmten Verfahrweges umsetzt. Da die Objektbewegungen in ihrer zeitlichen Reihenfolge gespeichert werden, können sie in einem Wiedergabemodus in einer Animation wieder abgespielt werden. Darüber hinaus können die Punktfolgen (Positions- und Orientierungswerte) durch Interpolations- und Approximationsverfahren in eine parametrische Bahnrepräsentation überführt werden und in symbolischer Form (als Roboterprogramm) an einen externen Robotersimulator übergeben werden.

4.6 Kombination des geometrischen Modellierers mit Simulationskernfunktionen

Nachdem die grundlegende Funktionalität des geplanten Modelliersystems durch Abarbeiten der Schritte 1-5 verfügbar war, konnte die zentrale Frage der Kopplung des Modellierers mit Simulationswerkzeugen bearbeitet werden. Hierfür kamen zunächst zwei verfügbare Simulatoren in Betracht. Zum einen der Roboter-Simulator COSIMIR, der benutzt wurde, um (1) das Konzept des Aufzeichnens von dynamischen Vorgängen (Bewegungen) zu überprüfen und (2) um Möglichkeiten zur Generierung von Steuerprogrammen aus vorgemachten Handbewegungen abzuschätzen. Zum anderen wurde eine Schnittstelle zu einem ereignisorientierten Materialflußsimulator (GHOST) realisiert, welcher im Rahmen eines studentischen Projektes entstand.

Die Kopplung mit einem Simulator für Roboter-Arbeitszellen

Ein Schwachpunkt bisher vorhandener Konzepte zur Eingabe von Programmen innerhalb von Offline-Programmiersystemen für Roboter ist die Eingabe dreidimensionaler Bewegungsdaten mit den zweidimensionalen Eingabegeräten Maus oder Grafiktablett und die Darstellung der Daten auf dem Bildschirm. Diese Vorgehensweise erfordert ein häufiges Wechseln der Ansichten, um Kollisionen zu erkennen und um eindeutige räumliche Positionsbestimmungen vornehmen zu können. Darüber hinaus bleibt oft ein hohes Maß numerischer Eingaben erforderlich, die zu einer weiteren Erhöhung des Zeitaufwandes beitragen. An diesen Schwächen herkömmlicher Simulatoren und Offline-Programmiereinrichtungen setzt die Programmierung durch gegenständliches Vormachen

an. Der Mensch verfügt über ein hochentwickeltes Vermögen zur Auge-Hand-Koordination, das Aufgaben trivial erscheinen läßt, welche sich bei der Roboterprogrammierung als schwierig erweisen. Komplexe Abläufe in einer Roboterzelle können mit Hilfe eines Datenhandschuhs effizient eingeben und zu Steuerprogrammen weiterverarbeitet werden. Gut ausgestattete Robotersimulatoren können unter Verwendung von Präprozessoren verschiedene Programmiersprachen verarbeiten, das Verhalten der Robotersteuerungen einschließlich inverser Kinematik nachbilden und die errechneten Winkel und Längendaten der Roboterachsen anhand eines virtuellen geometrischen Modells visualisieren. Durch das Abarbeiten der eingegebenen Programme werden Fehler im Ablauf deutlich. Mögliche Fehler sind Kollisionen, Positionsungenauigkeiten, nicht erreichbare oder unzulässige Achsstellungen und singuläre Punkte in der Kinematik. Diese können mit Programmeditoren korrigiert werden. Auf diese Weise läßt sich die Sicherheit und Verwendbarkeit der Roboterprogramme soweit erhöhen, daß ihre Inbetriebnahme nur noch einen geringen Zeitaufwand erfordert. Um die beschriebenen Vorteile zu demonstrieren, wurde ein Robotersimulator an das virtuelle Datenmodell des Modelliersystems angekoppelt.

Als Robotersimulator kommt das System COSIMIR des Instituts für Roboterforschung der Universität Dortmund zum Einsatz (Breuer et al 1994). Dieses erscheint aufgrund einer universellen ASCII-Schnittstelle zur Roboterzellenbeschreibung, die dokumentiert ist, und wegen seiner verschiedenen Präprozessoren für Roboter-Programmiersprachen, gut zur Ankopplung an eine Umgebung mit automatisch generierten Daten geeignet.

Das zur Handhabung der geometrischen und dynamischen Daten sowie zur Visualisierung eingesetzte Virtual-Reality-Tool „World Tool Kit“ der Firma Sense8 verwendet eigene Konventionen zur Repräsentation von Geometrie-, Bewegungs- und Darstellungsdaten. Geometriedaten werden im ASCII-Format „NFF“ (Neutral File Format Version 2.0) dargestellt. Dieses erlaubt die Beschreibung dreidimensionaler Körper durch Polygone, deren Flächen mit den Attributen Farbe, Textur und anderen optischen Eigenschaften versehen werden können. Zum Zweck der Simulation müssen die statischen und die dynamischen Elemente in ihrer Ausgangsposition beschrieben werden. Die Art und Detaillierung der Geometriebeschreibung hängt von ihrer Relevanz für den Simulationsablauf, den Erwartungen an die grafische Darstellung und vom Simulationskonzept ab. Während bei ereignisgesteuerten Simulatoren häufig Bitmaps für die Visualisierung eingesetzt werden, erfordern geometrisch anspruchsvolle Systeme, wie z.B. Robotersimulatoren, eine dreidimensionale vektorielle Beschreibung der Geometrie.

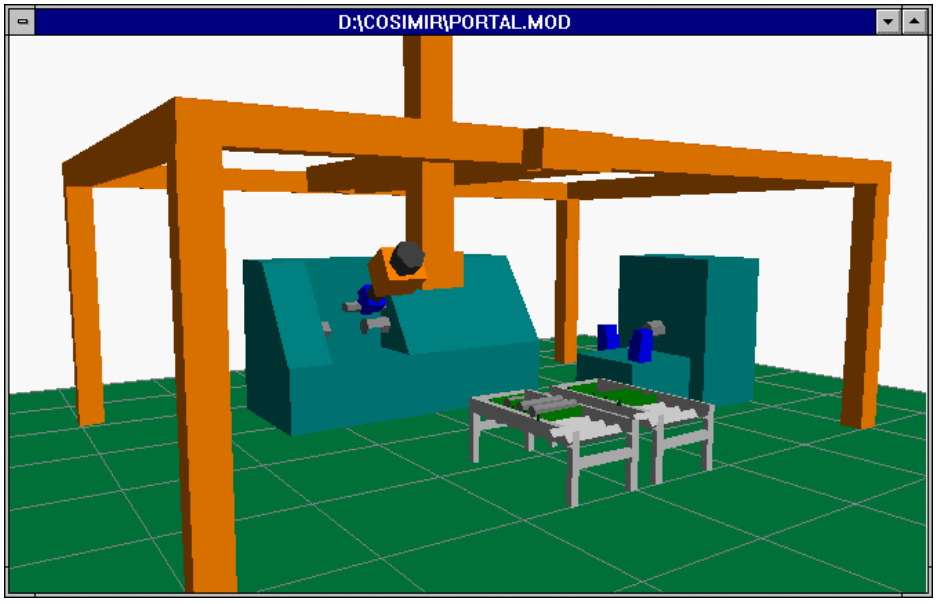

Abbildung 3: Eine Arbeitszelle mit Roboter in COSIMIR

Die Beschreibung der Arbeitszelle enthält die Darstellung des Roboters, der beweglichen (handhabbaren) und der unbeweglichen Objekte. Außerdem sind eine Reihe technologischer Parameter erforderlich, um das Verhalten und die Fähigkeiten des realen Roboters zutreffend zu spezifizieren. Der Simulator COSIMIR enthält einen Editor, mit dem Roboterzellen interaktiv definiert bzw. konkretisiert werden können. Für die hier dargestellten Zwecke wurden vorgefertigte Robotermodelle aus der Bibliothek von COSIMIR benutzt. Abbildung 3 zeigt dafür ein Beispiel.

Auf dem im gegenständlichen Modellierer existierende virtuelle Abbild des Modellszenarios basierend, wird automatisch eine Modelldatei für den Robotersimulator erzeugt. Die Arbeitszelle des Roboters wird in COSIMIR mit der Sprache EGEMOL beschrieben, die in einer Backus-Naur-Form dokumentiert ist (vergl. EGEMOL 1995). Diese „MOD-Dateien“ lassen sich mit den entwikkelten Konvertierungsfunktionen generieren. Der Roboter selbst ist nicht in die virtuelle Welt des Prototypen integriert, da der Benutzer die Transportvorgänge in der Modellwelt mit der Hand vormacht und damit in dieser Phase den Roboter ersetzt. Erst für die Simulation in COSIMIR wird ein Roboter aus der Bibliothek hinzugefügt (Abbildung 4). Abbildung 5 zeigt die Verknüpfung der Komponenten zur MOD-Datei.

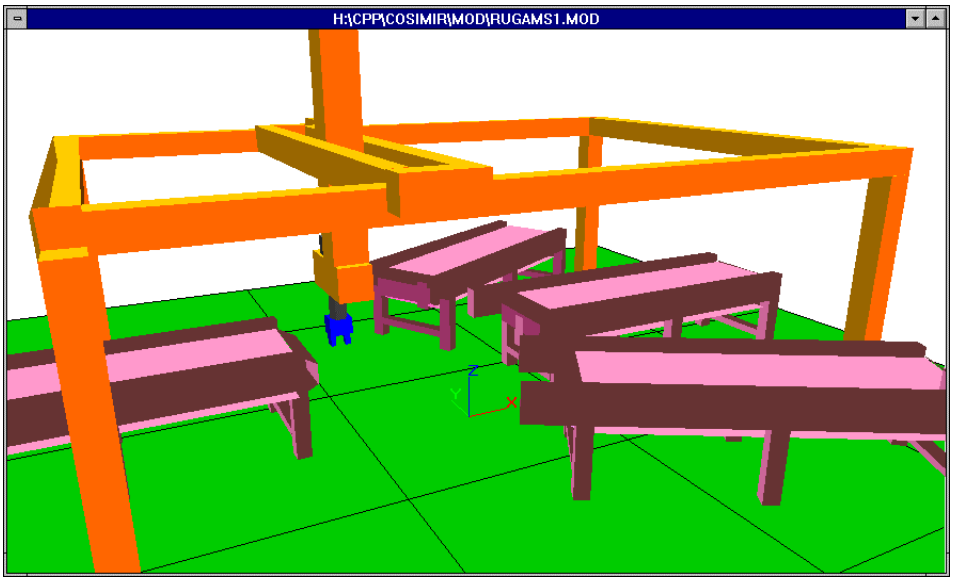

Abbildung 4: Darstellung des automatisch generierten Förderbandszenarios in COSIMIR

Pick and Place-Aufgaben werden durch die Wege beschrieben, die von der Hand mit gegriffenen Objekten zurückgelegt und vom Trackingsystem aufgezeichnet werden. Der Anfang dieser Wege wird durch das Greifen, das Ende durch das Loslassen eines Objektes erkannt. Die Informationen über Objektbewegungen werden in relativen oder absoluten Bewegungsvektoren gespeichert. Diese bestehen aus Komponenten zur Positions- Orientierungsbeschreibung. Sequenzen aus absoluten oder relativen Positionsvektoren werden als Pfade bezeichnet. Im Modellierer werden diese geteachten Pfade als Attribute von passiven beweglichen Objekten gespeichert. Aus diesen wird das Arbeitsprogramm des Roboters generiert.

Durch eine lineare Interpolation und das Überschleifen der Zwischenpunkte fährt der simulierte - und anschließend ein realer - Roboter die vorgemachten Pfade in einer Annäherung nach. Die Genauigkeit hängt hierbei von der Anzahl der Punkte, der Verfahrgeschwindigkeit und der Genauigkeit beim Überschleifen ab. Einige Steuerungen erlauben außerdem die Verbesserung der Bahngenauigkeit durch eine zirkulare Interpolation.

Zur Übertragung der Bewegungsdaten vom Modellierer zum Simulator und von dort auf die Steuerung eines Roboters, ist eine geeignete Programmiersprache auszuwählen. COSIMIR unterstützt die Programmiersprachen Mitsubishi Robot Language (MRL), die Bewegungs- und Ablauf-Programmiersprache (BAPS) und die Industrial Robot Language (IRL). Für den Transfer der Pro-

gramme nach COSIMIR wurde die Sprache IRL ausgewählt. Sie ist herstellerunabhängig und ist seit 1992 in der DIN 66 312 dokumentiert. Durch die Normung besteht ein klar abgegrenzter und gut spezifizierter Sprachumfang, was eine Grundlage für die automatische Programmerzeugung ist. Als Schnittstelle zwischen dem Modellierer und COSIMIR wurde das ASCII-Dateiformat „WTK“ entwickelt. Abbildung 5 stellt den Datenfluß der Schnittstelle schematisch dar.

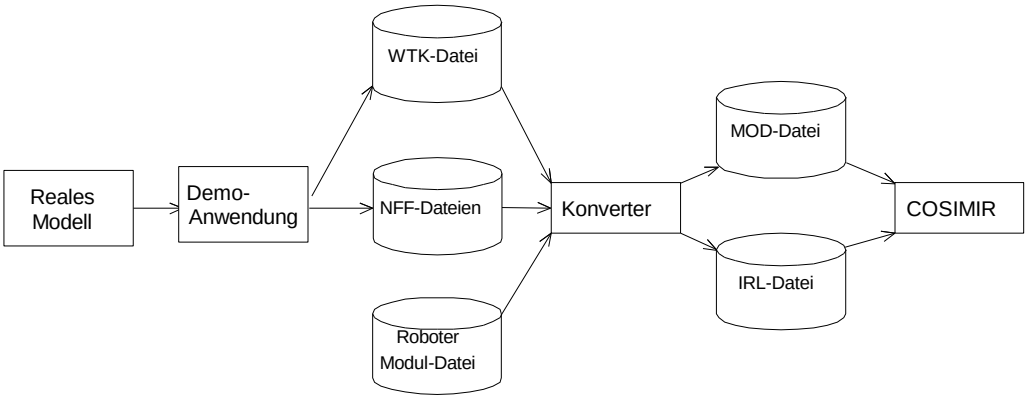

Abbildung 5: Anbindung des Robotersimulators COSIMIR an die Demonstrationsanwendung

Die Anbindung des Robotersimulators COSIMIR zeigt die prinzipielle Verwendbarkeit der Daten aus dem virtuellen Modell für die Beschreibung von Simulationsmodellen und zur Programmierung von Fertigungseinrichtungen.

Die Kopplung mit GHOST

Das zentrale Anwendungsgebiet für das im Projektverlauf entstehende Modelliersystem ist die Planung und Steuerung von Materialströmen in produktionstechnischen Anlagen. Neben der Focussierung auf einzelne Geräte wie Roboter, ist insbesondere das Zusammenwirken von Einzelkomponenten zu untersuchen. Dieser Problematik wird heute mit spezialisierten Simulationswerkzeugen entgegengetreten. Für die Kopplung von gegenständlichen Modellen mit Simulatoren ist es notwendig, eine Zuordnung zwischen den Objekten der beiden Repräsentationsformen herzustellen. Mit der Anbindung von GHOST, wird zunächst folgendes Phasenkonzept unterstützt:

- Modellierung: Auf dem Modelliertisch wird aus Bausteinelementen ein Simulationsszenario zusammengesetzt. Ausgehend von einem Initialzustand, bei dem rechnerinterne und externe Bausteine über ihre räumlichen Koordinaten einander zugeordnet sind, kann durch Bewegen und Rotieren der Elemente mit der Hand ein geometrisch-strukturelles Modell einer Anlage aufgebaut werden. Es entstehen gleichzeitig zwei Modelle: ein gegenständliches auf dem Tisch und ein virtuelles im Modellierer
- Datenübertragung an den Editor: Das virtuelle Modell wird nach seiner Fertigstellung an einen Simulations-Editor übergeben, indem eine Kommunikationsverbindung (auf Sockets basierend, vgl. Abschn. 4.9 „dynamischer Datenaustausch“) zwischen dem Editor und dem Modellierwerkzeug hergestellt wird. Es erfolgt eine Datenübertragung vom Modellierer zum Editor, so daß der Editor Informationen über die Position und Orientierung der verwendeten Modellelemente erhält.
- Topologieerkennung: Mit Hilfe von Suchalgorithmen des Editors werden logisch-funktionale Verbindungen zwischen den atomaren Modellbausteinen hergestellt. Ketten von Förderbändern werden gebildet, Eingänge werden mit Ausgängen verbunden, Material fließt von Quellen zu Senken.
- Verfeinerung des Modells: Die Modellbausteine besitzen zwar generell voreingestellte Werte, die ihr Verhalten spezifizieren. So haben z.B. Förderbänder einen vorgegebenen Wert für das Attri-

but „Geschwindigkeit“, aber um reale Situationen abzubilden, sind diese jeweils den gegebenen Umständen anzupassen. In dieser Phase können auch Fehler, die bei der Topologieerkennung aufgetreten sind, korrigiert werden.

- Datenübertragung an den Modellierer: Das verfeinerte Modell wird wiederum mittels einer Kommunikationsverbindung an den Modellierer übertragen. Der Modellierer übernimmt jetzt die Funktion, das Simulationsszenario realitätsnah, dreidimensional visuell darzustellen. Hierfür wird auf das World Tool Kit zurückgegriffen, das auch die interaktive Veränderung von Sichtperspektiven auf des grafische Modell erlaubt und das Auswählen von Objekten mit durch Anklicken ermöglicht.
- Simulation des Modells (zur Zeit in Arbeit): Das Modell, das sich sowohl im Editor als auch für Zwecke der Visualisierung im Modellierer befindet, wird mit Hilfe der Simulationsfunktionen von GHOST simuliert. Über die Kommunikationsschnittstelle erhält der Modellierer die Daten über sich dynamisch verändernde Objekte. So wird z.B. die Bewegung von Paletten in einem Transportsystem vom Simulator berechnet und die veränderten Koordinaten an den Modellierer weitergeleitet. Eine schnelle Abfolge der Koordinatenverschiebung ermöglicht eine flüssige dreidimensionale Animation der Modelldynamik.

Zusätzlich zum Datenaustausch über eine online Kommunikation zweier Prozesse, ist die Speicherung von Modelldaten in einem persistenten Datenformat in Arbeit. Damit können Simulationsszenarien dauerhaft gespeichert und bei Bedarf wieder bearbeitet werden.

4.7 Konzeptentwicklung für eine akustische Rückkopplung

Ein wichtiger Aspekt des Modellierens im Realen ist die Hinwendung der Benutzer zu den physikalischen Modellbausteinen während des Modellierens. Damit einher geht die Abwendung vom Bildschirm und von der konventionellen Benutzungsoberfläche mit visuellen Rückkopplungsmechanismen. Um dem Benutzer eine erfolgte Selektion oder Deselektion (Greifen, Loslassen) eines Objektes zu signalisieren, wurde eine akustische Rückkopplung implementiert. Bereits einfache Töne, die unmittelbar nach der Erkennung eines Griffes (oder des Loslassens) generiert werden, geben dem Benutzer die notwendige Sicherheit, daß seine Aktionen vom Rechner erkannt wurden. Die akustischen Signale unterteilen das Handeln des Modellierenden in unterscheidbare Sequenzen: das Betrachten und Diskutieren eines Modellzustandes bis zum aktiven Handeln, ein Objekt wird gegriffen, ein Signal ertönt, das Objekt wird bewegt - mit dem Wissen des Benutzers, daß die Bewegung aufgezeichnet wird - und wieder abgesetzt und losgelassen, wiederum von einem akustischen Signal begleitet. Es ist ein neuer Modellzustand hergestellt.

Die bisherige Erfahrung hat gezeigt, daß bereits ein einziges Signal zur Separation von Handlungssequenzen ausreicht. Die handelsübliche Hardware für die Erzeugung der akustischen Signale genügt diesen einfachen Anforderungen. In weiteren Experimenten kann untersucht werden, ob die akustische Ausgabe komplexer Informationen in Form von generierter Sprache am Bildschirm textlich/grafisch dargestellte Inhalte ersetzen oder ergänzen kann.

4.8 Entwicklung der Bausteine für den Prototypen: Förderbandelemente, Werkstückträger

Für die prototypische Anwendung des Förderbandsystems wurden gegenständliche Förderbandmodelle aus Fischertechnik Bausteinen erstellt. Die Verwendung von Fischertechnik hat den Vorteil der flexiblen Verwendbarkeit der Bausteinelemente und der vielfältigen Möglichkeiten zur Motorisierung und Sensorisieung der Modelle, welches interessante Optionen für weitergehende Untersuchungen sind. Für Zwecke der Fabriksimulation ist Fischertechnik ein weit verbreitetes und anerkanntes System. Als Paletten- bzw. Containermodelle dienen verschieden farbige Holzklötze. Die Transportbänder wurden als Vorbild für die ebenfalls fertiggestellten geometrischen Modelle verwendet, die aus Gründen der Komplexitätsminimierung aus wenigen Polygonen und Kanten beste-

hen (siehe Abb. 2). Die Anzahl der Polygone im virtuellen Modell hat einen maßgeblichen Einfluß auf die Geschwindigkeit bei der räumlichen Darstellung des virtuellen Modells. Grafische Abbildungen realer Anlagen mit hohem Ähnlichkeitsgrad sind nur mit spezieller Beschleunigungs-Hardware effizient handhabbar.

Ein weiteres Baukastensystem wurde aus Holz angefertigt. Es besteht aus den Elementen Förderband, Quelle, Senke, Knotenpunkt, Puffer und Bearbeitungsstation. Durch den im Vergleich zu Fischertechnik-Modellen etwas kleineren Maßstab lassen sich damit auch komplexere Anlagen aufbauen.

4.9 Systemstruktur

Die entwickelte Software ist in Module aufgeteilt. Grundlegende Maßgabe der Programmierung ist die Wiederverwendbarkeit einzelner Module, denen jeweils bestimmte Aufgaben zugeordnet werden. Im Mittelpunkt steht das „Universum“ (die Begriffswahl erfolgte in Anlehnung an die WTK-Terminologie). Darin sind die eigentlichen Abläufe des Modellierens mit der Modellierumgebung manifestiert. Es dient als Entwicklungsrahmen (Framework) für konkrete Anwendungen, wie dem Förderband-Szenario. An das Universum sind die anderen Module angegliedert, es steuert den Datenfluß, verwaltet Ereignisse und reagiert auf Eingaben, die von Sensoren registriert oder über Dialoge erfaßt werden. Die Menge aller Module ergibt die gesamte Funktionalität des Modellierers. Da mit realen, greifbaren Modellen gearbeitet wird, wurde dieser Real Object Modeller (ROM) genannt.

Abbildung 6 zeigt die Struktur des ROM. Die zum Modellieren verwendeten Modelle werden in einer Klassenstruktur hierarchisch definiert. Das Definieren von Modellklassen mit der Zuordnung von Attributen etc. ist unabhängig von konkreten Anwendungen. Die Instanzen der definierten Klassen werden zur Laufzeit des Programms erzeugt und in das Universum eingefügt. Dynamische Vorgänge mit Objekten des Universums werden durch die *Regelerkennung* erkannt. Beispielsweise wird die Bewegung von Objekten des Typs „Palette“ als Umsetzprozeß in einem Materialfluß interpretiert. Die Erkennung von Gesten und Griffen erfolgt mit Hilfe des Moduls *Gestenerkennung*. Wird ein Griff erkannt, so wird ein entsprechenden Ereignis erzeugt, worauf die Anwendung zu reagieren hat. Die Gestenerkennung erhält Daten von *Gerätetreibern*, die die Datenkommunikation mit der Spezial-Hardware (5^{th} *Glove* und *Polhemus* Tracking System) steuern.

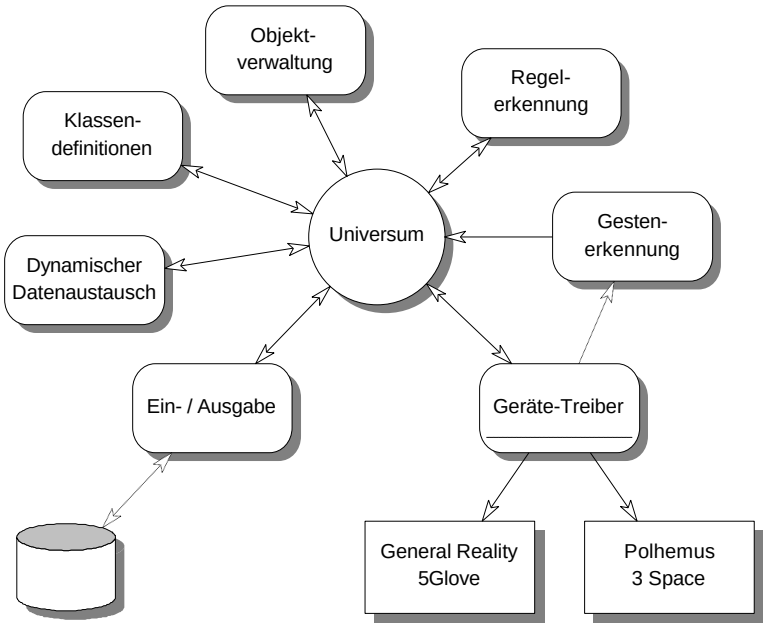

Abbildung 6: Module des Real Object Modellers (ROM)

Der RO-Modellierer arbeitet ausschließlich mit Objektgeometrien, die u.a. visuell dargestellt werden können. Zusätzliche Attribute werden von externen Programmen, wie Simulatoren oder Editoren, verwendet. Mit einer Schnittstelle zum *dynamischen Datenaustausch* können Modelldaten zur Laufzeit synchronisiert werden. Das Laden von Modelldaten in den Modellierer aus Dateien erfolgt mittels eines *Ein-/Ausgabemoduls*. Dafür wurde ein spezielles Datenformat konzipiert.

4.10 Systemeinsatz und Systemumgebung

Das Modelliersystem läuft auf Standard-PCs mit MS Windows ab der Version 3.1 (mit Win-Sock-Bibliotheken). Als Entwicklungstool wurde zunächst der C/C++ Compiler von Borland in der Version 4.52 eingesetzt. Später wurde aber aus Gründen besserer Kompatibilität zu neueren Windows-Systemen und zum World Tool Kit Microsoft's Visual C++ Compiler in der Version 4.0 verwendet. Als nützliches Entwicklungswerkzeug hat sich auch die Klassenbibliothek LEDA (Library of Efficient Datatypes and Algorithms) des Max-Planck Institutes für Informatik, Saarbrücken erwiesen.

Folgende PC Konfiguration wird gegenwärtig benutzt:

- Pentium PC 100 MHz
- 32 MB Hauptspeicher
- Diamond Stealth Grafikkarte (2MB)
- Soundkarte
- Schnittstellenkarte für zusätzliche serielle Ports
- Windows NT & Windows 95

Eine detaillierte Systembeschreibung mit der Dokumentation der Klassenstruktur ist in einem Systemhandbuch enthalten (Faust, 1996).

5. Weitere Vorgehensweise

Gegenständliches *geometrisches* Modellieren wird inzwischen auch wissenschaftlich breiter vertreten. Fitzmaurice et al (1995) verwenden mehrere sensorisierte Bausteine, um damit auf einer Displayfläche die Geometrie von Freiformkurven beidhändig zu manipulieren. Murakami & Nakajima (1995) verwenden einen Orts- und Dehnungsmeßwürfel, um Freiformflächen zu gestalten. Beide Ansätze sind durch ihre Abhängigkeit von speziellen Eingabegeräten beschränkt. Die Verwendung eines universellen Sensorhandschuhs erweist sich als vielseitiger. Andererseits bietet die Sensorisierung der Realobjekte den Vorteil, auch eine Fremdeinwirkung auf das Interface im Rechner abbilden zu können. Sensorisierte Hände verlangen, daß alle Veränderungen der Realwelt ausschließlich mit diesen Händen erfolgen, damit die Ähnlichkeit zwischen realer und virtueller Welt erhalten bleibt. Es ist zu erwarten, daß die drei grundsätzlich unterscheidbaren Ansätze: Sensorisierung der Hand (Ursache), Sensorisierung der Objekte (Wirkung) und Sensorisierung einer unbeteiligten Instanz (Beobachter) ihre jeweiligen Anwendungsfelder haben werden und daß es auch für die Kombination aller drei Konzepte eine Berechtigung gibt. Im weiteren Verlauf des Projektes sollen zunächst die Möglichkeiten des ersten Konzeptes weiter untersucht werden.

Vereinfachung der Grifferkennung durch Kontextorientierung

Bezüglich der Grifferkennungsproblematik hat sich gezeigt, daß eine Behandlung als allgemeines Mustererkennungsproblem mit möglichst hoher Auflösung der Muster (Fingerkrümmung aller Finger) nicht vorteilhaft ist. Als effektiver erwies sich der Ansatz, die Griffvielfalt einzuschränken und

dem Anwender bestimmte wohlunterscheidbare einfache Grifftypen vorzuschreiben. Diese Einschränkung erwies sich bisher im Modellierungsprozeß nicht als störend.

Beidhändiges Modellieren

Nach den positiven Erfahrungen des einhändigen Modellierens deuten sich vielfältige Möglichkeiten des zweihändigen Arbeitens an. Dabei sind nicht nur Konzepte von Interesse, bei denen beide Hände auf einer semantischen Ebene operieren, wie zur Erstellung von Freiformen, Topologien, dynamischen Regeln, Synchronisationen, sondern auch solche, bei denen die eine Hand zeigende Funktionen, wie identifizieren, selektieren, separieren, übernimmt, während die andere Hand operativ tätig ist und formt, bewegt, fügt, trennt.

Konsequentere Vermeidung von Medienbrüchen

Der existierende Prototyp ermöglicht das Modellieren mit der sensorisierten Hand, verlangt aber noch verschiedene Eingaben über die Tastatur. Insbesondere erfolgt die Variation des virtuellen Modells über die konventionellen Eingabemedien Tastatur und Maus. Dieser Medienbruch ist außerordentlich störend (Benutzer bedient Maus mit angezogenem Sensorhandschuh). In der Weiterentwicklung soll ein einfacherer Wechsel zwischen Realität und Virtualität möglich sein: mit dem Handschuh kann auf realen und virtuellen Objekten operiert werden. Dazu muß der Bildschirm als eine neue Objektklasse mit Kontroll- und Sichtfunktionen eingeführt werden.

Bidirektionale Modellierung

Der existierende Prototyp unterstützt die Wirkungsrichtungen:

$$\text{Modellierer} \rightarrow \text{reales Modell} \rightarrow \text{Rechnermodell}$$
$$\text{und}$$
$$\text{Modellierer} \rightarrow \text{Rechnermodell.}$$

Durch die Ausstattung der realen Modelle mit Sensoren und Aktoren und deren Rechneranbindung kann eine Wirkungskette

$$\text{Modellierer} \rightarrow \text{reales Modell} \rightarrow \text{Rechnermodell} \rightarrow \text{Realität}$$

aufgebaut werden. Diese Schließung der Wirkungskette ermöglicht dann das gegenständliche Vormachen dynamischer Abläufe und deren direkte reale rechnergesteuerte Reproduktion.

Einsatz eines professionellen Simulators

Die experimentelle Kopplung zwischen dem Modellierer dem Simulator GHOST, der nur einfache Simulationsmodelle verwalten kann, wird anhand eines kommerziellen Produktes überprüft. Dafür wird der Simulator Simple++ verwendet. Das erweiterbare objekt-orientierte Konzept von Simple++ unterstützt die modulweise Zusammensetzung von Modellen und entspricht somit der Philosophie des gegenständlichen, baukastenbasierten Modellierens.

6. Publikationen, Workshops und Kontakte

Das Projekt RUGAMS und damit das Konzept des synchronen Modellierens im Realen und Virtuellen, Real Reality (RR) wurde mehrfach präsentiert und in Veröffentlichungen dargestellt:

- im November 1995 auf dem 7. GI Workshop Hypermedia und KI in Hannover, der Beitrag von W. Bruns und V. Brauer mit dem Titel „Greifendes und Begreifendes Modellieren im Realen und Virtuellen“ wurde veröffentlicht in: Neugebauer & Wiesener (Hrsg.), FORWISS-Report zum 7. Workshop Hypermedia & KI, FORWISS, München, Mai 1996.
- vom 22. bis 27 April 1996 wurde das Projekt auf einem Stand auf der Industriemesse in Hannover präsentiert. Dort wurde ein Prototyp des Modelliersystems ausgestellt und das Konzept mit Vertretern der Industrie und Forschung diskutiert.
- im Mai 1996 hat V. Brauer einen Vortrag auf dem 2. IFIP 5.10 Workshop on Virtual Prototyping in Arlington, Texas gehalten. Der Beitrag wird veröffentlicht unter W. Bruns & V. Brauer, Bridging the Gap between Real and Virtual Modeling - A New Approach to Human-Computer Interaction, Proc. of the 2. IFIP 5.10 Workshop on Virtual Prototyping, Arlington, Tx, 1997. Eine Vorab-Version des Beitrags ist als artec Paper Nr. 46 erhältlich.
- im Oktober 1996 erschien ein Fernsehbericht über das Projekt - im Auftrag von 3-Sat wurde ein Beitrag zur Wissenschaftssendung HITEC gedreht.
- im November 1996 erschien von V. Brauer ein Artikel in der Fachzeitschrift ACM SIGGRAPH Computer Graphics, Vol. 30, No. 4, mit dem Titel „Simulation Model Design in Physical Environments“.
- im Dezember 1996 erscheint von W. Bruns in der Zeitschrift AI & Society ein Artikel mit dem Titel „Grasping, Communicating, Understanding - Connecting Reality and Virtuality“. Dieser Beitrag ist vorab als artec Paper Nr. 44 erhältlich.

Die Anwesenheit auf der Messe bzw. den Workshops hat interessante Kontakte, Diskussionen, Anregungen und Ideen zur Folge gehabt. Insbesondere zur Klärung von Fachfragen sollen die Kontakte in Zukunft vertieft oder in Form von konkreten Kooperationen in einen produktiven Rahmen gestellt werden.

7. Literatur

Becker, B.-D. (1991): Simulationssystem für Fertigungsprozesse mit Stückgutcharakter - Ein gegenstandsorientiertes System mit parametrisierter Netzwerkmodellierung. Dissertation, Universität Stuttgart.

Brauer, V. (1994): Feature-basierte Erkennung dynamischer Gesten mit einem Datenhandschuh. Diplomarbeit, Universität Bremen

Brauer, V. (1996): Gestenerkennung mit einem Datenhandschuh, artec Paper Nr. 42, Universität Bremen, artec

Breuer, S.; Scharf, S. (1994): Thorausch, M.: COSIMIR, Softwarepaket zur zellorientierten Simulation und Programmierung von Industrierobotrn, Festo Didactic, Esslingen

Bruns, F. W. (1993): Zur Rückgewinnung von Sinnlichkeit - Eine neue Form des Umgangs mit Rechnern. Technische Rundschau Heft 29/30, S. 14-18

Bruns, F. W., Heimbucher, A., Müller, D. (1993): Ansätze einer erfahrungsorientierten Gestaltung von Rechnersystemen für die Produktion. artec-Paper 21, Bremen

Bruns, F. W., Brauer, V. (1996): Greifendes und begreifendes Modellieren im Realen und Virtuellen. 7. GI-Workshop „Hypermedia und KI". Hannover, Nov. 95

Bruns, W., Brauer, V. (1996): Bridging the Gap between Real and Virtual Modeling - A New Approach to Human-Computer Interaction -. IFIP-Workshop „Virtual Prototyping“, Texas (im Druck, auch als artec Paper Nr. 46, Universität Bremen)

EGEMOL (1995): Einführung in die Arbeitszellenmodellierung mit EGEMOL, Vorabdruck: Institut für Roboterforschung, Universität Dortmund

Faust, Martin (1996): The Grasping Hand, Dokumentation zur Demonstrations-Software für das DFG-Projekt RUGAMS, Universität Bremen, artec

Fikes, R., Gruber, T., Iwasaki, Y. (1994): The Stanford How Things Work Project. Knowledge Systems Laboratory, Stanford University

Finin, T., Weber, J., Wiederhold, G., Genesereth, Fritzon, R., McGuire, J., Shapiro, S., Beck, C. (1993): DRAFT Specification of the KQML Agent-Communication Language. University of Maryland Baltimore County, Baltimore

Fishwick, P. A., Luker, P. A. (1991): Qualitative Simulation Modeling and Analysis. Springer Verlag, New York

Fitzmaurice, G. W., Ishii, H., Buxton, W. (1995): Bricks: Laying the Foundations for Graspable User Interfaces. CHI 95 Mosaic of Creativity, May 7-11, 1995, S. 442-449

Foley & Van Damme (1990): Computer Graphics: Principles and Practice, Addison-Wesley

Fuss, H. (1994): Wie simuliert man am bequemsten halbgeordnete Prozesse?! In: ASIM Mitteilungen aus den Arbeitskreisen, Heft 40, TU Wien, S. 93-96

Goldberg, A., Robson, D. (1983): Smalltalk-80. The language and it's implementations. Addison-Wesley, Reading, Mass.

Gordon, G. (1981): GPSS Session. In: Wexelblat, R. L. (Hrsg): History of Programming Languages. Academic Press, New York, S. 403-426

Gruber, T. R. (1992): A Translation Approach to Portable Ontology Specifications. Knowledge Systems Laboratory, Stanford University, Technical Report KSL 92-71

Gruber, T, Gautier, P. (1993): Machine-generated Explanations of Engineering Models: A compositional modeling approach. Proc. of the 1993 Int. Joint Conf. on Artificial Intelligence

Hommel et al (1994): The TU Berlin High Precision Data Glove, Forth International Workshop on Work with Display Units (WWDU), Milan, October 1994

Isdale, J. (1993): What is Virtual Reality - a homebrew Introduction. Isdale Engineering

Iwasaki, Y., Chandrasekaran, B. (1992): Design Verification Through Function and Behavior-Oriented Representations: Bridging the Gap Between Function and Behavior. Proc. of the Second Int. Conf. On Artificial Intelligence in Design

Iwasaki, Y., Vescovi, M., Fikes, R. (1994): A Causal Functional Representation Language with Behavior-Based Semantics. Knowledge Systems Laboratory, Stanford University, KSL-94-10

Kang, S. B., Ikeuchi, K. (1994): Grasp Recognition and Manipulative Motion Characterization from Human Hand Motion Sequences. Proc. of IEEE Int. Conf. on Robotics and Automation, San Diego, Cal., Vol 2, S 1759-1764

Klußmann, J. (1994): Aufbereitung von CAD-Zeichnungsdateien zur weiteren Verwendung in Materialflußsimulatoren unter Berücksichtigung der STEP-Norm. Diplomarbeit, Universität Bremen

Krauth, J., Meyer, R. (1993): Vergleich von Simulationsergebnissen verschiedener Simulatoren anhand eines Beispielmodells. In: Sydow, A. (Hrsg.): Tagungsband zum 8. ASIM-Symposium Simulationstechnik in Berlin. ASIM-Reihe "Fortschritte in der Simulationstechnik", Band 6, Braunschweig - Wiesbaden: Vieweg Verlag

Krueger, M. (1991): Artificial Reality II. Addison-Wesley Pub.

Kuhn, A., Reinhardt, A. (1993): Handbuch Simulationsanwendungen in Produktion und Logistik. Vieweg, Braunschweig

Kuipers, B. (1994): Qualitative Reasoning. Modeling and Simulation with Incomplete Knowledge. The MIT Press, Cambridge, Massachusetts

MacKenzie, C., Iberall, T. (1994): The Grasping Hand. Amsterdam: Elsevier

Noche, B., Wenzel, S. (1991): Marktspiegel Simulationstechnik in Produktion und Logistik. Verlag TÜV Rheinland, Köln

Nygaard, K. (1981): SIMULA Session. In: Wexelblat, R. L. (Hrsg): History of Programming Languages. Academic Press, New York, S. 439-480

Reinhardt, A. Kühne, K. (1988): Modellbausteine und Werkzeuge für den Anlagenbau - Entwurf, Dimensionierung und Angebotserstellung. In: Ameling, W. (Hrsg.): Simulationstechnik, 5. Symp. Simulationstechnik, Aachen, S. 464-469

Schäfer, K. (1995): Objektorientierte Modellierung zur Simulation des Steuerungsverhaltens von modularen Transfersystemen. Diplomarbeit, Universität Bremen

Scharf, P., Spies, W. (1990): Fabriksimulation - Ergebnisse einer Befragung von Anwendern. VDI-Z 11, S. 62-65

Scheel, J., Bruns, F. W., Busekros, L., Frank, G., Heimbucher, A., Hofferberth, D. (1994): Simulation von Arbeit und Technik - Entwicklung, Implementation und Betrieb von Programmsystemen für Planung, Simulation und Animation im Fabrikbereich an der Hochschule Bremen, Abschlußbericht, PSA-Institut, Hochschule Bremen

Splanemann, R. (1995): Teilautomatische Generierung von Simulationsmodellen aus systemneutral definierten Unternehmensdaten. Dissertation, Universität Bremen

Sturmann D. (1992): Whole-Hand-Input, PhD Thesis, Media Arts and Sciences, MIT

Sturmann D. & Zeltzer D. (1994): A Survey of Glove-Based Input Devices, IEEE Computer Graphics and Applications, Vol. 14, No. 1, Jan. 1994

Warnecke, H.-J., Bullinger, H.-J. (Hrsg.) (1994): Virtual Reality '94 - Anwendungen & Trends. Springer-Verlag, Berlin

Wenzel, S. (1993): Objektorientierter Software-Baukasten zur Konfiguration von Simulationswerkzeugen. Informationstechnik und Technische Informatik, 35, 6, S. 25-33

Zeigler, B. P. (1990): Object Oriented Simulation with Hierarchical, Modular Models: Intelligent Agents and Endomorphic Systems. Academic Press, New York

Zühlke, D., Küster, J. (1994): Die Simulation als Planungshilfsmittel. VDI-Z 136, 5, S. 33 - 37

Zweiter Zwischenbericht zum DFG Forschungsprojekt

RUGAMS

Prof. Dr.-Ing. F. Wilhelm Bruns
Universität Bremen
Forschungszentrum Arbeit und Technik (artec)
Dezember 1997

Forschungsvorhaben:
Rechnergestützte Übergänge zwischen gegenständlichen und abstrakten Modellen produktionstechnischer Systeme (RUGAMS)

DFG Gz.: BR 1556/2-2

Berichtszeitraum:
Januar bis Dezember 1997

Projektgruppe:
Prof. Dr.-Ing. F. Wilhelm Bruns
Dipl.-Inform. Volker Brauer
Dipl.-Ing. Kai Schäfer
Cand.-Inform. Martin Faust
Dipl.-Ing. (FH) Wolfgang Tieben
Dipl.-Ing. (FH) Joachim Hinrichs

Universität Bremen
Forschungszentrum Arbeit und Technik (artec)
Bibliothekstr. (MZH)
28359 Bremen

Tel.: 0421-218-4206 (-2435, Sekretariat)
Fax: 0421-218-4449
email: bruns@artec.uni-bremen.de

Zweiter Zwischenbericht zum DFG Forschungsprojekt RUGAMS

Der vorliegende Zwischenbericht dokumentiert den Fortschritt des Projekts im zweiten Projektjahr, Stand Dezember 1997.

1. Arbeitspakete

1.1 Weiterentwicklung des Prototypen

Während des Berichtszeitraums stand die Weiterentwicklung des im ersten Projektjahr entstandenen Prototypen einer gegenständlichen Modellierungs-Umgebung im Mittelpunkt. Hierfür wurden umfangreiche Arbeiten an dem äußeren Erscheinungsbild, den Interaktionsmöglichkeiten mit den Gegenständen und der internen System-Architektur vorgenommen. Anhand eines Beispiel-Szenarios wird der aktuelle Stand der Entwicklung dargestellt.

1.1.1 Anwendungsbeispiel

Als Beispiel haben wir aus dem Bereich der flexiblen Fertigung eine Anlage ausgewählt, welche von Krauth (1991) als Referenzanlage für den Vergleich von Simulationssystemen beschrieben wurde. Diese Anlagenkonfiguration bietet eine Reihe von Vorteilen für eine prototypische Realisierung:

- Es liegt eine genaue Beschreibung der Anlage und ihres Verhaltens vor. Die Unklarheiten der ursprünglichen Veröffentlichung werden durch Ergänzungen in EUROSIM 4, 1992 und Krauth (1995) weitgehend ausgeräumt.
- Es liegen zahlreiche simulierte Vergleichslösungen und Ergebnisse vor, die entweder in EUROSIM Newslettern veröffentlicht oder am Bremer Institut für Betriebstechnik und angewandte Arbeitswissenschaft (BIBA) erstellt wurden. Dort wurde ein detaillierter Vergleich der Lösungen mit verschiedenen Simulatoren vorgenommen. Dabei wurde eine Reihe von Ungenauigkeiten in der Aufgabenbeschreibung und Differenzen zwischen den Verhaltensweisen der Simulationsbausteine verschiedener Simulatoren, und somit in der simulierten Anlage, festgestellt und begründet.
- Die Anlage besteht lediglich aus 3 verschiedenen Bausteintypen: Förderband, Quertransport und Maschine. Durch die Spezifikation unterschiedlichen Steuerverhaltens und verschiedener Parameter dieser Elemente, kann eine Vielzahl von Problemen aus dem Bereich der flexiblen Fertigung untersucht werden.
- Die Anlagengröße erlaubt eine prototypische Realisierung, ist aber auch hinreichend komplex, um den Einsatz von Simulationstechniken zur Analyse zu rechtfertigen.

Im Einzelnen besteht die Anlage aus 8 gleich aufgebauten Stationen. Von einer Hauptstrecke können Paletten auf einen Nebenzweig, welcher zu einer Bearbeitungsstation führt, abgezweigt werden. Nach der Bearbeitung wird die Palette wieder in die Hauptstrecke eingeschleust (

Bild 1).

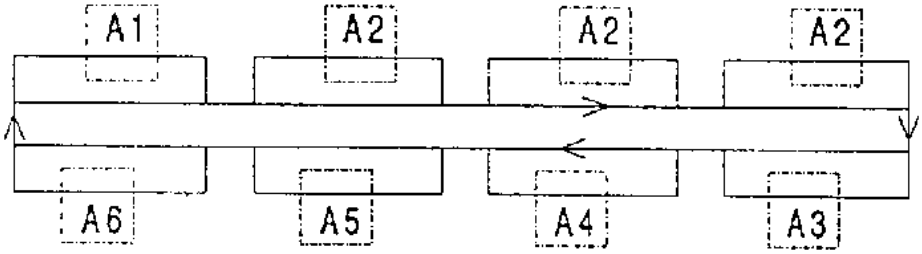

Bild 1.: Schema der Referenzanlage

Jede einzelne Station besteht aus 2 Quertransporten (S_x, S_y), 2 Förderbändern (B_1, B_2), die als Staustrecken dienen können, und einer Bearbeitungsstation (A_x) (Bild 2). Die Abmessungen der Paletten betragen 360 x 360 mm.

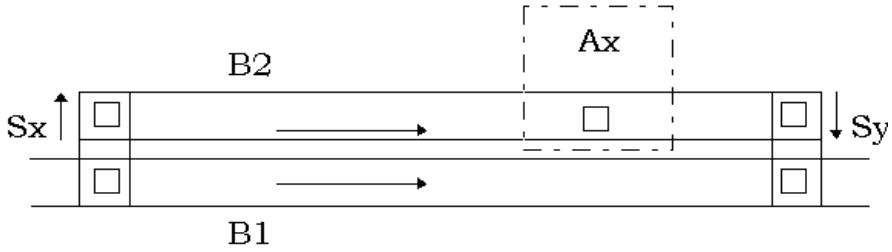

Bild 2.: Einzelstation

Station	Arbeitsgang	Bearbeitungs-zeit [s]
A1	Be- und Entladen	7,5
A2	P1	60
A3	P2	20
A4	P3	20
A5	P4	20
A6	P2-P4	30

Tabelle 3.: Arbeitsgänge in den Stationen

Arbeitsgang	Vorgänger
Be-/Entladen	Palette leer oder A1-A4
P1	WS unbearbeitet oder A2-A4
P2	
P3	
P4	

Tabelle 4.: Bearbeitungsplan

In den Stationen können die in Tabelle 3 angegebenen Arbeitsgänge (P_i) durchgeführt werden. Tabelle 4 enthält den Bearbeitungsplan für die Werkstücke (WS). Ein Nebenzweig, der zu einer Bearbeitung führt, darf nur befahren werden, wenn der Puffer vor der Bearbeitungsstation einen freien Platz aufweist. Auf dem Quertransport darf sich nur eine Palette zur Zeit befinden. Für eine genauere Beschreibung siehe Krauth (1991 u. 1992).

1.1.2 Aufbau der Anlage in der gegenständlichen Modellierungs-Umgebung

Entsprechend der vorgegebenen Referenzanlage, wurde ein Bausteinsatz aus Fischertechnik angefertigt, mit dem der Aufbau des Szenarios als Realmodell möglich ist. (Bild 3 und Bild 4). Zusätzlich zu diesen Bausteinen im kleinen Maßstab, wurden für einen Ausschnitt aus der Anlage – einer Nebenstrecke mit Bearbeitungsstation – funktionale Modelle im größeren Maßstab angefertigt. Dieses Modell wird an eine Steuerung angeschlossen, so daß eine Überprüfung des spezifizierten Anlagenverhaltens und der Steuerungsprogramme an einem Realmodell vorgenommen werden kann.

Zu den Realmodellen wurden virtuelle Modelle erstellt. Diese bestehen zum einen aus einem geometrisch konstruierten Abbild und zum anderen enthalten sie eine Reihe von Datenattributen, mit

denen das dynamische Verhalten der Bausteine beschrieben wird. Aufgrund dieser doppelten Repräsentation – der gegenständlichen und der virtuellen – nennen wir die zum Modellaufbau benutzten Bausteine auch Zwillings-Objekte („Twin-Objects“). Das Ergebnis dieser Arbeiten ist eine wiederverwendbare Bibliothek aus Twin-Objects (siehe Abschn. Bausteinbibliothek).

Als Arbeitsfläche für den Aufbau von Modellen wurde ein Modelliertisch angefertigt (Maße in cm: 200 x 120 x 100), um den sich 4 - 5 Personen komfortabel gruppieren können. Unter der Tischfläche befindet sich das Tracking-System, welches die Position der angeschlossenen Datenhandschuhe erfaßt und somit eine Analyse der Aktionen, die mit den Twin-Objets ausgeführt werden, erlaubt. Bild 3 zeigt den sukzessiven Aufbau eines Modells mit dem Datenhandschuh.

Bild 3: Aufbau eines Anlagenmodells. Mit beiden Datenhandschuhen können Twin-Objects aus der im Vordergrund dargestellten Object-Box entnommen und in das Modell eingefügt werden. In der Object-Box erhalten die gegenständlichen Modellbausteine eine definierte Initialposition, bevor sie in das Modell eingesetzt werden. Im Hintergrund ist ein virtuelles Abbild der Szene zu erkennen.

Die Eingabetechnik wurde inzwischen hardware-technisch erweitert und zu einer flexiblen Lösung ausgebaut. Zur Zeit können zwei Datenhandschuhe gleichzeitig betrieben werden. Diese Beschränkung besteht jedoch nur aufgrund fehlender zusätzlicher Anschlüsse und Datenhandschuhe. Das Software-Konzept für die Verwaltung der Datenhandschuhe und der Gestenerkennung erlaubt theoretisch den Betrieb von *n* Handschuhen, die an unterschiedlichen Rechnern angeschlossen sein können. Die Erweiterungen bezüglich der Grifferkennung und des dafür notwendigen Handmodells wird in einem gesonderten Abschnitt erläutert.

Neben der mehrhändigen Modellierung, wird auch die beidhändige unterstützt. Bild 4 zeigt, wie ein Benutzer einen linken und einen rechten Datenhandschuh trägt. Weiterhin ist ein fertig aufgebautes Modell der EUROSIM Anlage dargestellt. Aus der geometrischen Anordnung von Einzelelementen wird mit analytischen Methoden ein topologisch zusammenhängender Materialfluß ermittelt, welcher als Grundlage für die Generierung eines Simulationsmodells verwendet wird.

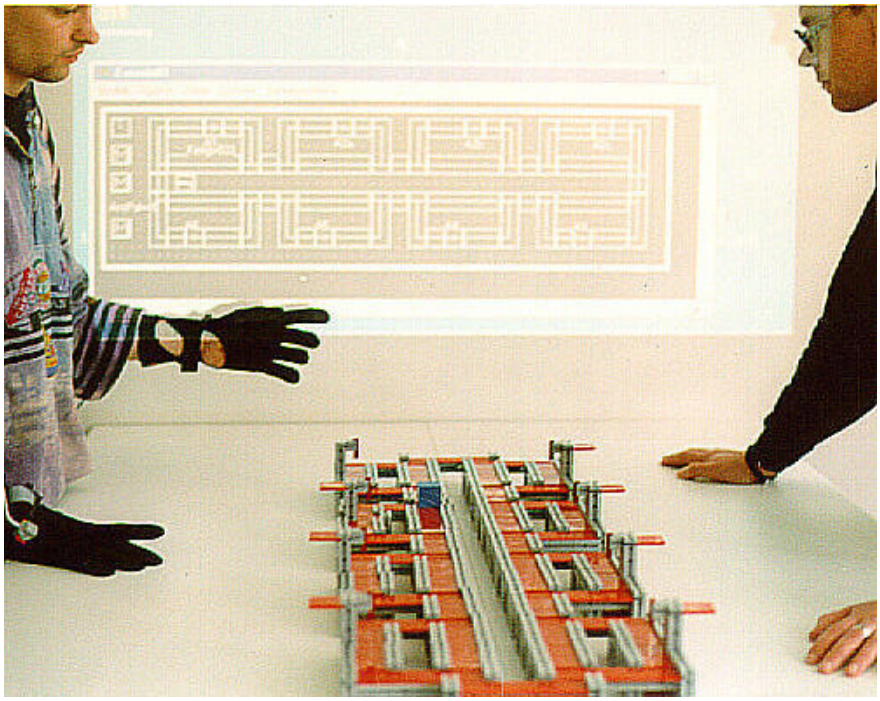

Bild 4.: Gegenständliches Modell der EUROSIM Anlage mit Simulationsmodell im Hintergrund.

Das Verfahren der Topologieanalyse basiert auf logischen Konstrukten, die dem virtuellen Teil der Twin-Objects zugeordnet sind. Jedes Objekt erhält als Attribut eine Liste von sog. *SensePoints*, welche virtuelle Anschlußpunkte zu benachbarten Objekte darstellen. Bei der automatischen Topologieanalyse wird davon ausgegangen, daß ein nahe bei einem Ausgangs-SensePoint gelegener Eingangs-SensePoint mit diesem als verbunden gelten soll. Die Begründung hierfür liegt im Verhalten von realen Funktionselementen. Wird eine Palette von einem aktiven Materialflußelement (Förderband) ausgegeben und befindet sich in unmittelbarer Nähe ein aufnahmefähiges Materialflußelement, findet ein Übergang statt, wie z.B. bei zwei in gleicher Richtung laufenden Förderbändern, die mit ihrem Ende bzw. Anfang unmittelbar aneinandergrenzen (vergl. Bild 5).

Für die Topologieanalyse wird dieses Prinzip genutzt, indem jeder Ausgangs-SensePoint mit allen Eingangs-SensePoints, die sich innerhalb einer definierten Umgebung befinden, verbunden wird. Die Zuverlässigkeit dieses Verfahrens ist abhängig von der Genauigkeit der Positionierung und wird daher nur teilautomatisch eingesetzt.

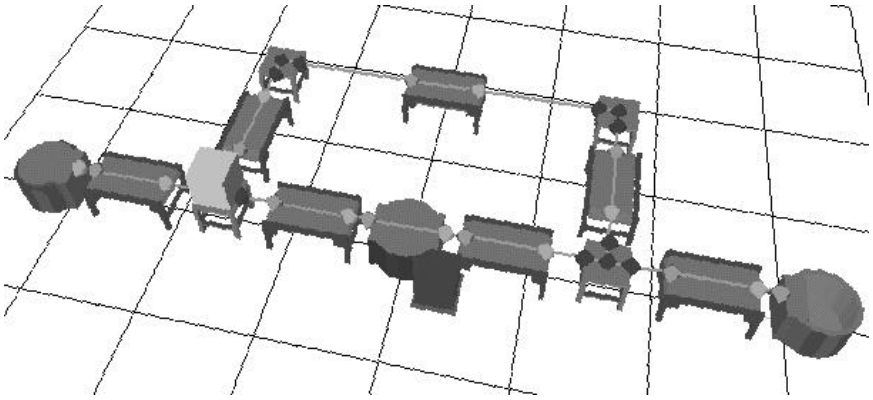

Bild 5: Analyse der Modelltopologie mit Hilfe von SensePoints. Die eingezeichneten Punkte markieren die Ein- und Ausgänge der Materialflußelemente. Die logische Verknüpfung der Einzelelemente kann algorithmisch bestimmt werden.

1.1.3 Überführung in ein Simulationsmodell

Zu jedem Realbaustein existiert ein Simulationsbaustein im Simulator Simple++. Zusammen mit der topologischen Verknüpfung und einem den Simulationsbausteinen zugeordneten Grundverhalten, welches den Materialfluß bestimmt, wird ein ablauffähiges dynamisches Modell erzeugt (Bild 6).

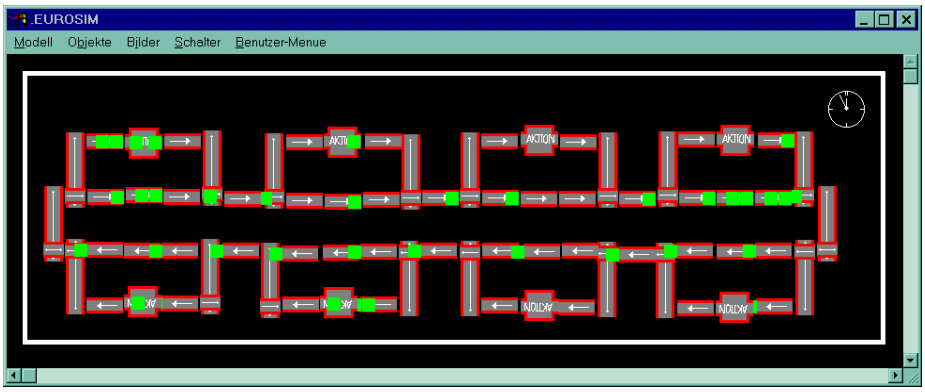

Bild 6: Simulationsmodell der EUROSIM Anlage. Aus der geometrischen Struktur des gegenständlichen Modells wird ein in Simple++ ablauffähiges Simulationsmodell erzeugt.

Der dynamische Ablauf des Simulationsmodells ist sofort sichtbar. Der Simulator generiert Paletten, schleust diese in das System ein und stellt deren Bewegungen auf den Materialflußelementen grafisch dar. Zusätzlich zu der schematischen Wiedergabe von Simple, wurde eine 3D-Visualisierung implementiert. Dafür werden die Geometriemodelle benutzt, die bereits für die Modellierung eingesetzt wurden. Die dreidimensionale Darstellung ist direkt mit der Simulation gekoppelt, d.h. der Fortschritt der bewegten Objekte in der 3D-Grafik entspricht exakt dem Modellzustand im Simulator (siehe Bild 7). Für diesen Zweck wurde ein Modul für den dynamischen Datenaustausch zwischen Prozessen realisiert (siehe Abschn. Systemarchitektur).

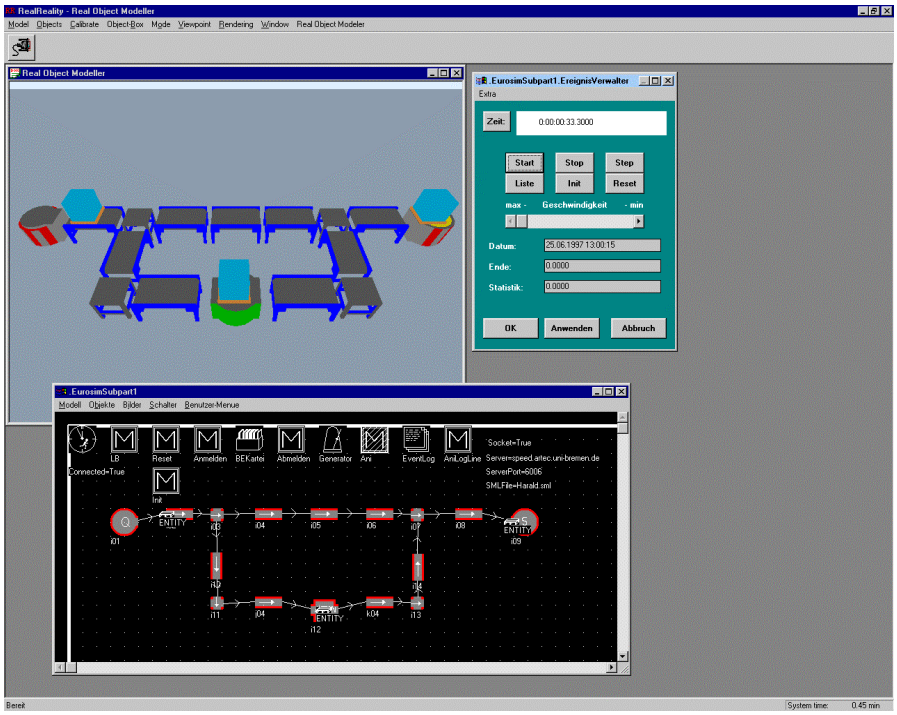

Bild 7: Ausschnitt aus der EUROSIM Anlage in Simple++ (Fenster unten) und die 3D-Visualisierung. Neben einer realistischeren Darstellung, kann zusätzlich in der 3D-Szene navigiert werden. Zudem können Informationen zu den Materialflußelementen abgerufen werden. Über den Ereignisverwalter (Fenster rechts oben) wird der Simulationslauf gesteuert.

In nachfolgenden Phasen der Modellierung wird das Simulationsmodell schrittweise verfeinert, indem das Standardverhalten redefiniert wird. Dafür wurden mehrere Konzepte zur Abstraktion von Regeln und Auflösung von Entscheidungskonflikten aus manuellen Benutzeraktionen entwickelt. Die ersten Ansätze dafür werden im folgenden erläutert.

1.2 Abstraktion dynamischen Verhaltens

Die bisher realisierte Kopplung zu Simulatoren mit vordefinierten Bausteinen wurde um die Möglichkeit erweitert, neue Bausteintypen durch Vormachen der Dynamik zu generieren. Die feste Vorgabe eines Bausteinverhaltens stellt eine Einschränkung des Konzepts der Modellierung im Gegenständlichen dar. Daher wurde für Bausteine eine Möglichkeit vorgesehen, das Verhalten, wie es im realen Modell spezifiziert wurde, zu übernehmen.

Für die bildschirmorientierte Eingabe existieren bereits verschiedene Konzepte für die Programmierung durch Vormachen (vgl. Programming by Demonstration, Cypher 1994). Das Vormachen an einem gegenständlichen Modell stellt jedoch eine große Weiterentwicklung und neue Perspektiven für dieses Konzept dar. Die mit der Hand vorgemachten Transportwege (Pfade) werden zunächst im Modell gespeichert. Durch hinzunehmen von Kontextwissen können diese Pfade interpretiert und Regeln generiert werden. Diese können später das Simulationsmodell oder eine reale Anlage über SPS steuern.

Grundlage für die Analyse ist die kontinuierliche Feststellung des nächstgelegenen Sensepoints zu der Palette, die aktuell mit dem Datenhandschuh bewegt wird. Findet ein Wechsel der größten Näherung einer Palette von einem Ausgangssensepoint zu einem Eingangssensepoint statt, wird dieser Übergang erkannt. Eine Strecke, die mit einer gegriffenen Palette auf dem Modell abgefahren wird, hinterläßt eine Spur gültiger Verknüpfungen (Bild 8, Tabelle 5). Redundante Übergänge werden ignoriert.

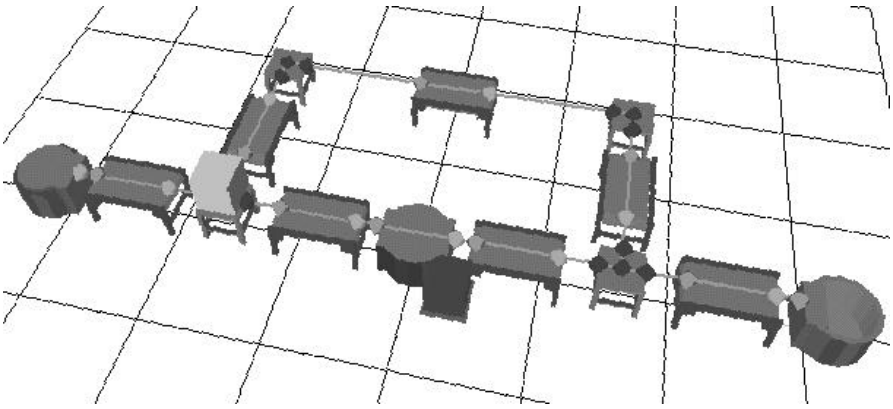

Bild 8.: Automatisch erkannte Verknüpfungen

Line	Class	Attribute	From	To	Number
1	ME_Palette	green	Source01.Sexit	Conv02.SEntry	1
2			Conv02.Sentry	Conv02.SExit	1
3			Conv02.Sexit	Node03.S1	1
4			Node03.S1	Node03.S	1
5			Node03.S	Node03.S2	1
6			Node03.S	Node03.S3	1

7			Node03.S2	Conv04.SEntry	1
8			Conv04. Sentry	Conv04.SExit	1
9			Conv04.Sexit	Action05.SEntry	1
10			Action05.Sentry	Action05.SExit	1
11			Action05.Sexit	Conv06.SEntry	1
12			Conv06.Sentry	Conv06.SExit	1
13			Conv06.Sexit	Node07.S1	1
14			Node07.S1	Node07.S	1
15			Node07.S	Node07.S2	1

Tabelle 5: Automatisch generierte Liste von Regeln

Wenn eine Palette von einem Ausgangssensepoint in verschiedenen Durchläufen zu unterschiedlichen Eingangssensepoints bewegt wird, liegt eine Verzweigung des Pfades und damit auch ein Konflikt vor (Tabelle 5, Zeilen 5 u. 6). Die verschiedenen Möglichkeiten an einer Verzweigung stellen einen Konflikt dar, der automatisch erkannt wird. Die gleiche Situation kann an einer Zusammenführung eintreten, wenn zwei Paletten gleichzeitig ankommen. Solche Konflikte können in Abhängigkeit von Palettenattributen oder von Anlagenzuständen aufgelöst werden.

Ein weiterer Kontext ist die Einbeziehung des Bearbeitungszustands von Paletten. Abhängig vom Bearbeitungszustand, soll eine Palette zu den Stationen geroutet werden, die noch ausstehende Arbeitsgänge fertigstellen können. Der Bearbeitungszustand einer Palette wird hierbei durch das Palettenattribut *Farbe* repräsentiert. Gemäß dieses Kontextes, wird eine Tabelle (wie z.B. Tabelle 6) generiert, die später als Entscheidungstabelle im Simulator das Verzweigungsverhalten steuert.

.Node03				
	Class	**Pal. Attribute**	**add. Cond.**	**Route To**
Default	Palette			S2
Default	Palette			S3
Rule 1	Palette	yellow	M1	S3
Rule 2	Palette	green		S2
Rule 3	Palette	green		S3
Rule 4	Palette	red		S2

Tabelle 6: Entscheidungstabelle zur Spezifikation von Verzweigungen

In der Praxis lassen Arbeitspläne häufig eine große Anzahl verschiedener Werkstückzustände zu. Diese sind nicht mehr sinnvoll durch unterschiedliche Farben darstellbar. Deshalb wurde ein Konzept zur Markierung von gegenständlichen Modellen entwickelt. Der Arbeitsplan einer Palette wird hierbei durch Marken, die durch das Attribut *Farbe* die verschiedenen Bearbeitungsgänge repräsentieren, abgebildet. Diese farbigen Marken werden wie bei den "Türmen von Hanoi" auf der Palette gestapelt. Abhängig von der Kombination erledigter und unerledigter Arbeitsgänge, kann die Konfliktsituation aufgelöst werden. Das Abarbeiten in der Station wird durch Entfernung der entsprechenden Marke von der Palette abgebildet.

Der Zustand eines Anlagenteils zum Zeitpunkt des Erreichens einer Verzweigung ist ein weiterer Kontext, mit dessen Hilfe Konflikte entschieden werden können. Würden beim Vormachen der Verzweigung die Zustände aller Elemente im Modell als Muster gespeichert, könnte später bei Übereinstimmung mit diesem Muster auf denselben Nachfolger umgelagert werden. Dieses Vorgehen ist jedoch unzweckmäßig. Zum Einen sind die zu speichernden Muster in großen Modellen sehr komplex, zum Anderen ist der größte Teil der hierbei zu berücksichtigenden Informationen

irrelevant[2]. Die Entscheidung wird deshalb nur von einer Auswahl von Sensepoints abhängig gemacht. Sensepoints können verschiedene Zustände eines Förderbausteins repräsentieren. Bei den beschriebenen Regeln spielen die Zustände

- Eingang frei,
- Palette austrittsbereit (Palette direkt vor dem Ausgang; Bearbeitung abgeschlossen) und
- Anlagenelement frei

eine Rolle. Optional könnten weitere Zustände, wie z.B. Element gestört, durch Marken auf zusätzlich zu vereinbarenden Sensepoints dargestellt und abgefragt werden. Eine die Anschaulichkeit des gegenständlichen Konzepts erhaltende Möglichkeit zur Kennzeichnung entscheidungsrelevanter Analgenzustände ist das Setzen von Marken auf die betroffenen Sensepoints.

Bild 9.: Konfliktentscheidung hängt vom Anlagenzustand ab

Im Modell wird eine Palette bis an den Sensepoint herangeschoben, der vor einer Verzweigung liegt. Danach werden alle entscheidungsrelevanten Sensepoints in ihrem gegenwärtigen Zustand mit Marken als relevant gekennzeichnet. Dann wird die Palette auf das gewünschte Nachfolgeelement geschoben. Der Computer speichert dieses Muster in Verbindung mit der überfahrenen Kante ab. Aus diesen Informationen wird Programmcode generiert, auf den in der Entscheidungstabelle als zusätzliche Bedingung verwiesen wird (Tabelle 6, Bild 10). Durch die Verknüpfung dieser Verhaltenssteuerung mit virtuellen Bausteinen entstehen Elemente neuen Typs.

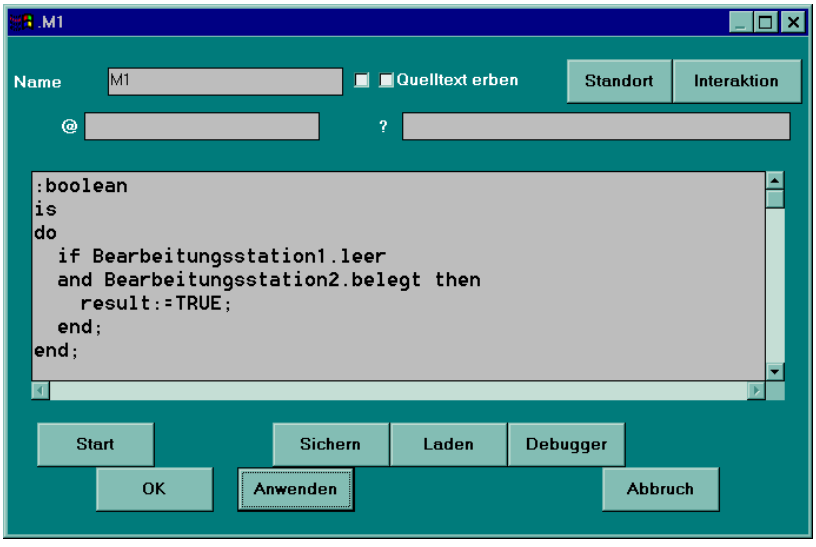

Bild 10.: Methode zur anlagenabhängigen Konfliktentscheidung

[2] Schon bei 10 Paletten auf 100 möglichen Sensepoints ergeben sich 100!-90!= 9,33*10^{157} mögliche Anlagenzustände.

1.2.1 Petri-Netze zur Darstellung des Steuerverhaltens

Petri-Netze bieten als Formalismus zur Darstellung von Anlagenmodellen und Steuerverhalten mehrere Vorteile (vgl. Schäfer 1995, Kämper 1991, Reisig 1985):

- Sie können automatisch aus dem virtuellen Modell des ROM und den analysierten Regeln generiert werden.
- Ihr Formalismus erlaubt die systematische Analyse zur Feststellung von Konflikten, Deadlocks, Kontaktsituationen (Sicherheit) und der Belebtheit des Netzes, welche über die einfache Suche nach Verzweigungen und Zusammenführungen hinaus geht.
- Petri-Netze haben eine Affinität zur Simulation und können direkt simuliert werden oder in Modelle für Materialfluß-Simulatoren umgewandelt werden. Die Simulation erlaubt die Visualisierung der Dynamik und die statistische Untersuchung des Anlagenverhaltens.
- Petri-Netze können sehr einfach in SPS-Programme übersetzt werden oder direkt als Steuerprogramm dienen (Aspern, 1993).

Was Petri-Netzen fehlt, ist z.B. die Abbildung des Stauverhaltens kontinuierlich laufender Förderbänder. Wir setzen daher eine Mischform aus Petri-Netzen und Materialflußbausteinen für die Simulation ein. Das Materialflußverhalten wird durch Förderbausteine abgebildet, die das Stauverhalten von Paletten korrekt wiedergeben. Das Steuerverhalten der Anlage wird durch Transitionen abgebildet, die das Anlagenverhalten beeinflussen. Diese Verfeinerungen des Simulationsmodells kann aus der Analyse des vorgemachten Verhaltens generiert werden. Bild 11a stellt ein einfaches Modell aus 5 Förderbändern dar. An der Verzweigung kommt das Defaultverhalten der Bausteine zur Anwendung: Es wird zufällig auf die beiden Nachfolger verteilt. In Bild 11b wurden durch Vormachen 2 Transitionen eingefügt, die das Verzweigungsverhalten beeinflussen: Wenn der obere Förderbaustein leer ist, wird nach oben abgezweigt (obere Transition), wenn der obere Förderbaustein belegt ist, wird auf den unteren Pfad weitergeleitet (untere Transition). An der Zusammenführung wurde eine Vorfahrtsregel für die von oben kommende Palette implementiert. Nur wenn am oberen Förderband keine Palette am Ausgang liegt, kann eine Palette von unten in die Zusammenführung laufen.

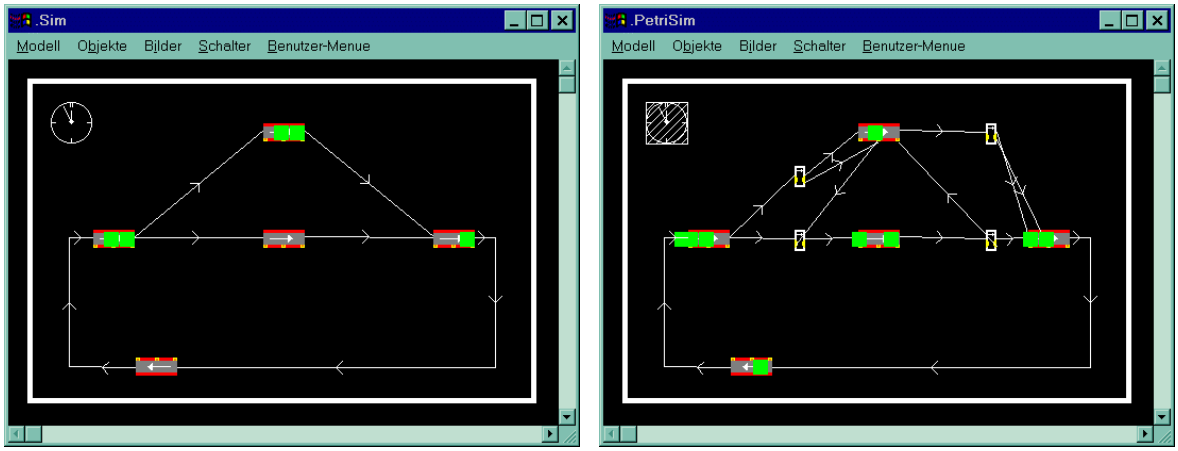

Bild 11.a, b: Petri-Netz gesteuertes Simulationsmodell

Informations- und Materialfluß haben unterschiedliche Eigenschaften. Paletten haben eine bestimmte Länge und können sich nur an einem Ort befinden. Informationen sind gleichzeitig im gesamten System verfügbar. Die Unterscheidung zwischen Material- und Informationsfluß wird durch entsprechende Signal- und Marerialfluß-Anschlüsse an den Förderbändern und Transitionen

vorgenommen. Gemischte Kanten zwischen Material- und Informations-Anschlüssen sind unzulässig.

Aus den Petrinetzen können SPS-Programmfragmente generiert werden. Wenn die Simulation ein befriedigendes Verhalten des Systems gezeigt hat, kann jede Transition als Anweisungsliste exportiert werden und an entsprechender Stelle in ein vorgefertigtes SPS-Programm integriert werden. Alle Bedingungen vor einer Transition müssen TRUE sein, alle Bedingung nach einer Transition müssen FALSE sein, nur dann ist der Rückgabewert der Anweisung TRUE. Hier die Anweisungsliste für die beiden Transitionen der Verzweigung:

```
LDN  %Foerderband1_belegt
S    %Transition_Verzweigung_oben

LD   %Foerderband1_belegt
S    %Transition_Verzweigung_unten
```

Diese Programmcodefragmente ergeben, in das vorgefertigte Programm eingesetzt, ein Steuerverhalten, das mit der Simulation übereinstimmt. Das Beschriebene Verfahren realisiert die Prozeßkette

RR-Modellierung → Simulation → SPS-Programm.

1.2.2 Abbildung des Steuerverhaltens als Anweisungsliste

Wenn ein sensorisiertes und aktorisiertes Modell die Zielanwendung darstellt, sind für dessen Steuerung SPS-Programme erforderlich. Um diese direkt und unabhängig vom Simulator generieren zu können, wurde eine Exportmöglichkeit der Regelerkennung für Anweisungslisten implementiert. Durch Konfiguration dieses Postprozessors kann die Anweisungsliste für verschiedene Steuerungen angepaßt werden. SPS-Programmiersysteme, die über eine Importmöglichkeit für Anweisungslisten verfügen (z.B. S5 für Windows der Gesellschaft für Automatisierungstechnik Beerfelden), können diese übernehmen, wodurch eine direkte Kopplung zwischen gegenständlicher Programmierung durch Vormachen und Steuerung eines Modells erreicht wird.

Zur Durchführung von SPS-Programmtests am Simulator wurde dieser um die Fähigkeit zur Simulation einer SPS erweitert. Für das von uns eingesetzte SIMPLE++ wurde ein Anweisungslisten-Interpreter entwickelt, der während der Simulation das Verhalten einer realen Siemens S5 SPS ereignisorientiert nachbildet. Der Detaillierungsgrad der Verhaltensbeschreibung von den verwendeten Materialflußbausteinen wurde soweit erhöht, daß die einzelnen Sensoren und Aktoren abgebildet werden. Simulationsläufe auf diesem Simulator führen zu zuverlässigen Aussagen über die Richtigkeit der Steuerprogramme. Die Prozeßkette

RR-Modellierung → SPS-Programm → Simulation / Steuerung

kommt mit weniger Zwischenschritten aus, als die im vorigen Abschnitt dargestelle, ist unabhängig von einem Simulator einzusetzen und integriert sich besser in die Schnittstellenstruktur heute verwendeter Anlagen. Da sich dieses Konzept als sehr nützlich und komfortabel für die Offlineprogrammierung erwiesen hat, wird es im nächsten Jahr im Rahmen dieses Projekts weiterentwickelt werden.

1.3 Erweiterung der Grifferkennung

Zur Verbesserung der Grifferkennung war zunächst vorgesehen, das kontextorientierte Konzept von MacKenzie & Iberall (1994) zu implementieren. Es beruht darauf, die Annäherungsphase an ein zu greifendes Objekt zu berücksichtigen, indem die Selektion des Zielobjektes durch den Richtungsvektor, welcher durch die Bewegung der Hand zum Objekt beschrieben wird, vereinfacht wird. Dieses Verfahren vereinfacht die Entscheidung, welches virtuelle Objekt dem real gegriffenem zuzuordnen ist. Die Analyse des bisherigen Verfahrens hat jedoch bezüglich der Objektselektion Schwachpunkte aufgedeckt, die mit der kontextorientierten Methode nicht entschärft werden können. In der Praxis ist es gelegentlich vorgekommen, daß statt des tatsächlich im Realen gegriffenen Modellbausteins im Virtuellen ein benachbartes Objekt der Hand zugeordnet wurde. Dieser Fehler resultierte aus den wenig detaillierten Objekt- und Handmodellen, die zur Grifferkennung verwendet wurden. Die Handposition wurde lediglich auf dem Handrücken gemessen und die Objektposition bezog sich ausschließlich auf den Objektmittelpunkt und nicht auf den gesamten Raum, den der Objektkörper beansprucht. Mit einem verfeinerten Hand- und Objektmodell wurde diese Fehlerquelle ausgeschlossen.

Mit dem verfeinerten Objektmodell wird nicht mehr die Distanz vom Tracking-Sensor der Hand zum Objektmittelpunkt ermittelt, sondern die kürzeste Verbindung vom Sensor zu Berührungspunkten auf der Objektoberfläche. Dies hat zur Folge, daß die Abmessungen des Objekts berücksichtigt werden müssen. Als Konsequenzen ergeben sich ein höherer Rechenaufwand und kompliziertere Algorithmen. Da wir bisher nur mit einfachen geometrischen Formen arbeiten, genügt es die umschließende Hülle eines Objektes (Bounding-Box), als Annäherung an die Objektform, in die Berechnung einzubeziehen (siehe Bild 12).

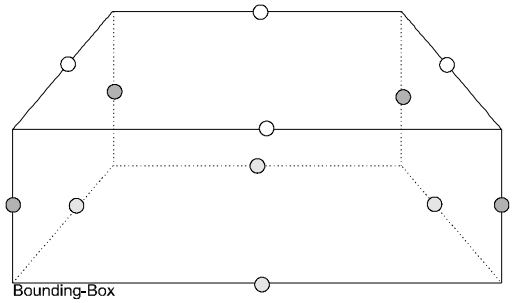

Bild 12: Bounding-Box mit zusätzlichen Berührungspunkten

Entlang der Kanten der Bounding-Box werden zusätzliche Punkte eingefügt, zu denen beim Greifen mit der Hand die Distanz gemessen wird. Damit wird eine wesentlich höhere Annäherung an die Objektgeometrie erreicht. Die 12 Berührungspunkte werden automatisch berechnet, so daß kein zusätzlicher manueller Modellierungsaufwand entsteht.

Weiterhin hat die Erfahrung gezeigt, daß die Messung der Handposition mit einem Sensor auf dem Handrücken nicht dem tatsächlichen Greifen und dem Einschließen von Objekten mit den Fingern nahe kommt. Daher wurde das einfache Handmodell, welches nur die Position und die Beugungswerte der Finger abbildet, um die Position der fünf Fingerspitzen erweitert. Diese Koordinaten werden approximativ aus der Handposition und den Beugungswinkeln der Finger berechnet. Bild 13 zeigt die Aufteilung eines Fingers in unterscheidbare Segmente. Da der verwendete Datenhandschuh pro Finger nur einen Beugungswert liefert, muß das Fingermodell vereinfacht werden.

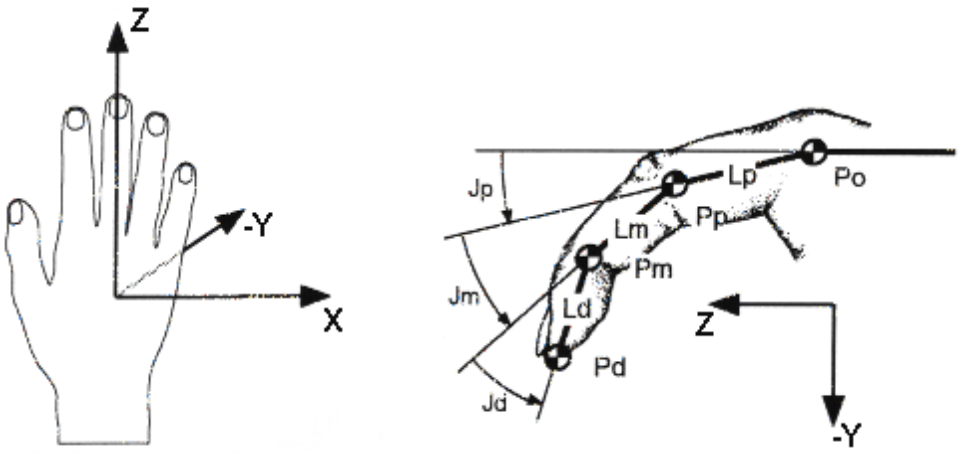

Bild 13: Segmentierung eines Fingergliedes zur Berechnung der Fingerspitzenposition

Die Vereinfachung beruht auf der Annahme, daß alle Segmente eines Fingers gleichmäßig gekrümmt werden. Die Winkel J_p, J_m und J_d werden aus einem einzelnen Beugungswinkel wie folgt berechnet: $J_p=J_m=J_d=J/3$.

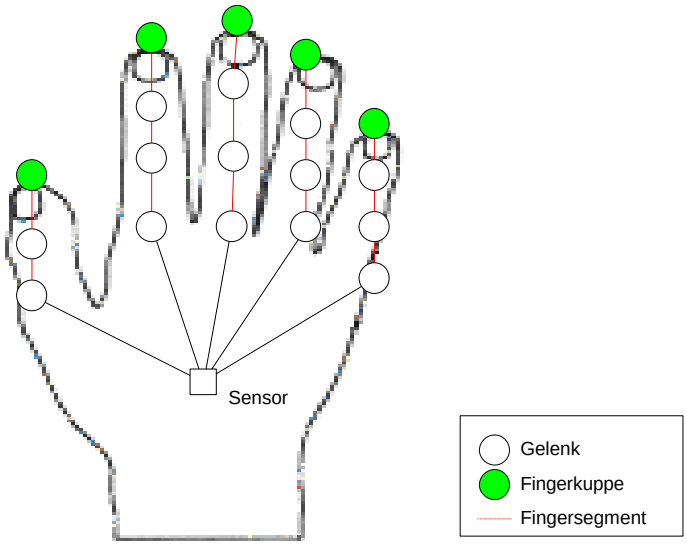

Bild 14: Erweitertes Handmodell. Zusätzlich zur Position des Sensors, werden auch die Fingerkuppen berücksichtigt Dadurch wird eine genauere Ortsbestimmung der Effektoren ermöglicht.

Da die anatomische Ausprägung der Hand individuell variiert, können benutzerspezifische Handmodelle in einer Datei gespeichert werden.

Mit den Maßnahmen zur Verfeinerung der Objekt- und Handmodelle wurde ein wesentlich zuverlässigere Erkennung von Griffereignissen erreicht. Die geschilderte Fehlerquelle wurde auf diese Weise beseitigt.

1.3.1 Beidhändigkeit

Um auch ein beidhändiges Modellieren zu ermöglichen, wurde die Software erweitert, mit der die Eingabegeräte (Datenhandschuhe) verwaltet werden. Dafür wurde eine Architektur implementiert, die den Anschluß mehrerer Handschuhe und die Erkennung individueller Griff- und Gestenmuster ermöglicht. Die zweihändige Benutzung - ein Benutzer trägt zwei Datenhandschuhe - wird praktisch eingesetzt, wenn der betreffende Benutzer

a) Objekte von einer Hand in die andere legt, um so einen größeren Aktionsradius zu erhalten, oder

b) zwei gleichzeitig gegriffene Objekte manipuluiert (bewegt) werden.

Im zweiten Fall kommt die Protokollfunktion des Objektmanagers zum Tragen. Das simultane Manipulieren zweier Objekte wird protokolliert, so daß in der anschließenden Analyse der Eingabeereignisse über den gesamten Zeitraum kausale Zusammenhänge zwischen räumlich-zeitlichen Objektmanipulationen hergestellt werden können.

1.4 Systemarchitektur des gegenständlichen Modellierers

Die Maßgabe für die Neukonzeptionierung der Systemarchitektur war, aus einem monolithischen, für einen einzelnen Zweck konzipierten Prototypen ein verteiltes, wiederverwendbares Software-System zu erstellen, das für eine Reihe potentieller Anwendungen einen Rahmen (Framework) bietet. Darin sind grundlegende Module, wie Gerätekontrolle, Netzwerkverbindungen oder Objektmanagement implementiert. Nach dem Vorbild des verbreiteten Client/Server Modells wird damit folgendes erreicht:

- Kommunikationsschnittstellen (Protokolle) regeln den Datenaustausch zwischen ggf. verteilten Prozessen. Die Ankopplung externer Anwendungen, wie z.B. Simulatoren wird standardisiert. Änderungen auf der Seite des Servers werden nur noch selten erforderlich.
- Durch die Verteilung verschiedener Prozesse erhöhen sich die Rechenkapazität und andere Ressourcen (z.B. serielle Schnittstellen).
- Eine logische und physikalische Trennung der funktionalen Komponenten (z.B. Grifferkennung, anwendungsspezifische Funktionen oder Visualisierung).

Mit dem Anspruch, gegenüber konkreten Anwendungen nur „Services“ anzubieten, die auf verschiedenen Rechnern verteilt werden können, steigen die Anforderungen an das Software-Engineering. Nur mit abstrakten Datentypen und Steuerungskonzepten kann das Ziel, wiederverwendbare Software-Bausteine zur Verfügung zu stellen, erreicht werden. Bild 15 zeigt die modulare Struktur des entwickelten Real Object Modellers.

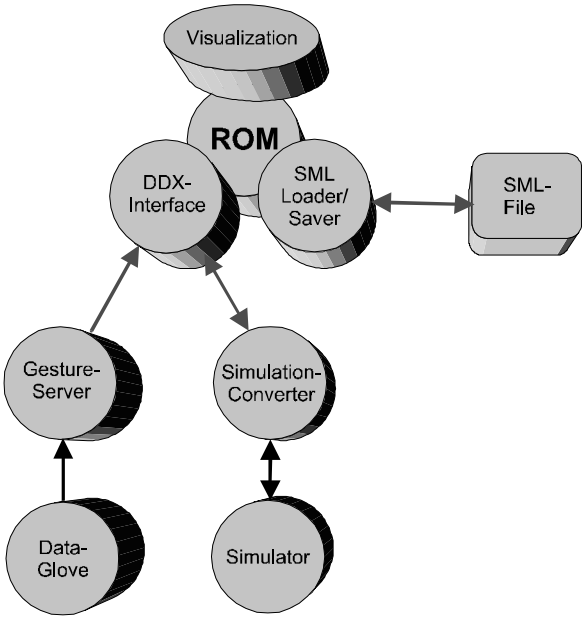

Bild 15: Systemarchitektur des Real Object Modellers. Im Mittelpunkt steht das Modul für die Objektverwaltung, Real Object Manager (ROM). An diesen schließen sich die weiteren Komponenten an.

Die Hauptaufgabe des Real Object Modellers besteht in der Verwaltung des Modells bzw. der darin enthaltenen Objekte. Die virtuellen Objekte werden von einer zentralen Instanz, dem Real Object Manager, verwaltet. Dieser kommuniziert auf zwei Wegen mit externen Prozessen und Programmen: zum einen dynamisch über spezielle Protokolle, welche den Nachrichten- und Datenaustausch regeln, und zum anderen statisch mittels Modelldateien, für die ein Datenformat (SML) entworfen wurde. Die dynamische Kommunikation nutzt die DDX-Schnittstelle (Dynamic Data Exchange) zum ROM. DDX basiert auf dem TCP/IP Standard und erlaubt somit auch den Datenaustausch zwischen vernetzten Rechnern. Dies wird zur Zeit für den Anschluß eines Gesten-Servers an das System praktiziert. An diesen Server ist die gesamte Hardware für die Handeingabe angeschlossen. Weiterhin laufen dort auch die Prozesse für die Erkennung von Griff- und Gestenereignissen, die an den ROM weitergeleitet werden. Diese werden dort interpretiert und anwendungsspezifische Reaktionen können ausgelöst werden. Des weiteren ist an die DDX-Schnittstelle ein Konverter, angeschlossen, mit dem die notwendigen Transformationen für die Simulation durchgeführt werden. Dieser Datenaustausch verläuft in beiden Richtungen, d.h. ein angeschlossener Simulator kann das Objektmodell im ROM nicht nur lesen sondern auch verändern. Für Simple++ wurde ein Protokoll implementiert, mit dem die von Simple++ generierten Ereignisse während eines Simulationslaufs an den ROM übertragen werden[3]. Dieses „Durchschleifen“ der Dynamik ist die Grundlage für die 3D-Visualisierung und Animation des Simulationsmodells. Wie dem Bild 15 zu entnehmen ist, ist an den ROM ein Visualisierungs-Modul angegliedert. Dieses ist mit Hilfe des Word Tool Kit implementiert, welches durch die Verwendung des OpenGL Grafik-Standards eine sehr leistungsfähige, realitätsnahe Grafikausgabe bietet. Für das nächste Projektjahr ist geplant, die Visualisierung auch über die DDX-Schnittstelle anzuschließen, so daß der Modellzugriff und die Grafikdarstellung über einen Client im Netzwerk möglich ist.

1.4.1 Simulation Modelling Language (SML)

SML ist eine objekt-orientiere Beschreibungssprache für Szenarien, die unter Anwendung der gegenständlichen Modellierungs-Umgebung aufgebaut wurden. Die Objekte dieser Szenarien können mittels einer hierarchischen Klassenstruktur deklariert werden. Gemäß dem objekt-orientierten Prinzip bilden die SML-Klassen den Rahmen für Attributzuweisungen, während Instanzen - als Ableitungen dieser Klassen - die in einer Szene enthaltenen Objekte darstellen. Die Trennung von Klassen und Instanzen erlaubt die Wiederverwendung der Klassen für verschiedene Anwendungen. SML, als ASCII Dateiformat, ermöglicht zudem das Laden und Speichern von Modellen, inklusive ihrer dynamischen Merkmale. Die Datei enthält auch die Historie der Änderungen, die während der Modellierung an den Objekten vorgenommen wurden. Damit erfüllt SML auch eine Protokollfunktion, denn die Historie kann mit einem Player/Recorder geladen und wieder abgespielt werden. Für das Abspielen wird wiederum die Visualisierung genutzt, so daß eine realitätsnahe Wiedergabe des Modellierungs-Prozesses zur Verfügung steht. Nachfolgend werden die Grundlegenden Elemente von SML kurz dargestellt.

1.4.1.1 Klassen

Eine Klasse definiert die Attribute der zu beschreibenden Objekte, wobei für den Real Object Modeller nur eine Menge von Standard-Attributen benötigt wird. Zudem können benutzerdefinierte Attribute in abgeleiteten Klassen hinzugefügt werden, wodurch das Dateiformat sehr flexibel an spezifische Anwendungsfälle angepaßt werden kann. In SML ist eine Reihe von Variablentypen vordefiniert: BOOLEAN, INTEGER, REAL und STRING. Aufbauend auf diesen Basistypen können, ähnlich wie in einer Programmiersprache, abstrakte Datentypen deklariert und in der Klassendefinition benutzt werden. Des weiteren gibt es reservierte Variablen, die für die Objektverwaltung benötigt werden:

[3] Eine weitere Verbindung zu einem Simulator wird im Rahmen einer Diplomarbeit realisiert. Hierbei handelt es um einen ereignisgesteuerten Simulator, welcher während eines studentischen Projekts entwickelt wurde.

- GEOMETRIE bildet das Attribut Geometrie ab, das für eine Boundary-Representation benutzt wird. Die Oberfläche von virtuellen Objekten wird durch externe Dateien vom Typ NFF, DXF oder 3DS beschrieben.
- SENSEPOINT definiert die oben erwähnten sensitiven Punkte, die u.a. für die Erkennung der Modelltopologie benötigt werden. Jeder SensePoint wird durch seinen Namen, Typ und Position beschrieben. Ein SensePoint ist entweder vom Typ Eingang, Ausgang oder typenlos.

Ein kurzes Beispiel illustriert die Klassendeklaration eines Förderbandes und die Ableitung einer Unterklasse mit dem Schlüsselwort „PARENT“.

```
CLASS Conveyor
BEGIN
      REAL Length
      REAL Width
      REAL Height
      REAL Speed
END_CLASS

CLASS Straight PARENT Conveyor
BEGIN
      Length = 37.0
      Height = 7.0
      Width = 9.0
      Speed = 2.0  % m/s (Kommentar mit Prozentzeichen)

      GEOMETRIE = "Straight.nff" SCALE 10.0
END_CLASS
```

1.4.1.2 Instanzen

Die Objekte einer Szene können auch als Instanzen bezeichnet werden. Ihre Eigenschaften sind in der Klasse, von der sie abgeleitet sind, deklariert. Eine Instanz fügt der Klassendefinition zwei weitere Variablen hinzu: POSITION und ORIENTATION. POSITION gibt die Position und ORIENTATION die absolute Orientierung der Instanz in einer aufgezeichneten Szene an. Der Bezugspunkt bei der Positionsangabe, die relativ zur Quelle des Trackingsystem angegeben wird, ist der geometrische Mittelpunkt eines Objekts. Die Arbeitsfläche (der Modelliertisch) hat die Höhe 0 (Y-Koordinate beim WTK). Die Orientierung der Objekte wird in Eulerwinkeln (ZYX, ±180.0°) angegeben. Wird ein Objekt bewegt, so wird der Instanz eine Variable vom Typ PATH hinzugefügt, womit der Bewegungsablauf während des Greifens beschrieben wird.

Beispiel:

```
INSTANCE IConveyor0 TYPE CStraight
BEGIN
  POSITION =    0.11   -3.65  -39.10
  ORIENTATION =    0.00    0.29    0.96
  PATH path1 = BEGIN
    TIME START = 83.570, END = 84.141
    NODES BEGIN
      0.000  54.000  -0.011  14.002 -176.065   0.028   0.023
      0.290  54.012  -0.001  13.956 -176.102   0.080   0.097
      0.571  53.983  -0.078  14.017 -176.082   0.001   0.088
    END     %NODES
    SENSEPOINTS BEGIN
      0.0 CONVEY28.Sausgang 5.971 64.0 0.0 14.0 175.0 0.0
```

```
        0.290 CONVEY30.Seingang 4.661 44.0 0.0 14.0 172.0 0.0
        0.571 CONVEY30.Sausgang 8.081 44.0 0.0 14.0 172.0 0.0
      END     %SENSEPOINTS
    END %PATH
  END_INSTANCE
```

Für das Einlesen und Parsen von SML-Dateien wurden die dafür notwendigen Programm-Module entwickelt und in einer C++-Bibliothek gespeichert. Dadurch können sehr einfach Post-Prozessoren erstellt werden, die SML-Dateien lesen und in andere Formate umwandeln. Der Konverter für Simple++ macht beispielsweise Gebrauch von dieser Funktionalität. Um das RR-Konzept als Frontend für andere Anwendungen zugänglich zu machen, die STEP-konformen Datenaustausch unterstützen, wurde ein EXPRESS-Exportfilter implementiert.

1.5 Bausteinbibliothek

Die Modellierung in der Real Reality Umgebung wird mit sogenannten Twin-Objects durchgeführt, Zwillingsobjekte, die gegenständlich im Realen und virtuell im Computer vorhanden sind. Für die Modellierung verschiedener Anwendungszwecke wurden unterschiedliche Bausteinsätze angefertigt.

Eine wichtige Eigenschaft der Twin-Objects ist ihre Größe und der Abbildungsmaßstab. Abhängig von der Komplexität der zu modellierenden Anlage, muß ein Maßstab gewählt werden, welcher den Aufbau aus handlichen Bausteinen auf der zur Verfügung stehenden Fläche des Modelltischs erlaubt. Zur Zeit werden drei Bausteinsätze verwendet. Einer zur Modellierung der gesamten EUROSIM-Anlage (Bild 4), die bei dieser Skalierung eine Länge von etwa 1 m hat. Ein zweiter Bausteinsatz bildet eine einzelne Fertigungszelle des EUROSIM-Modells in einem größeren Maßstab ab. Die einzelne Zelle hat im Modell ebenfalls eine Länge von etwa 1 m (Bild 16). Beide Bausteinsätze sind aus Fischertechnik Bauelementen zusammengesetzt (Bild 17 bis Bild 20), wobei der zweite mit Sensoren und Aktuatoren ausgestattet ist. Ein dritter Bausteinsatz ist aus Holz gefertigt und repräsentiert die Materialflußelemente auf einem höheren Abstraktionsniveau (Bild 9). Dadurch ist es möglich, eine größere Menge von verschiedenen Förderelementen durch die Bausteine darzustellen, was jedoch die Anschaulichkeit und Verständlichkeit mindert. Die Erfahrung hat gezeigt, daß detaillierte, dem Vorbild sehr ähnliche Abbildungen zu einer besseren Anschaulichkeit und Verständlichkeit für die Benutzer beitragen. Dadurch steigt jedoch der Aufwand für den Bau der Modellbausteine und sie sind weniger flexibel zu verwenden. Tabelle 7 zeigt die vorhandenen Bausteinsätze im Überblick (FMS steht für „Flexible Manufacturing System“).

	Bausteinsatz		
	Klötze	FMS klein	FMS groß
Material	Holz	Fischertechnik	Fischertechnik
Quelle	X	X	
Senke	X	X	
Förderb.	X	X	X
Knoten	X		
Quer-transp.		X	X
Maschine	X	X	X
Speicher	X		
Sensoren/Aktoren			X

Tabelle 7: Verfügbare Bausteinsätze

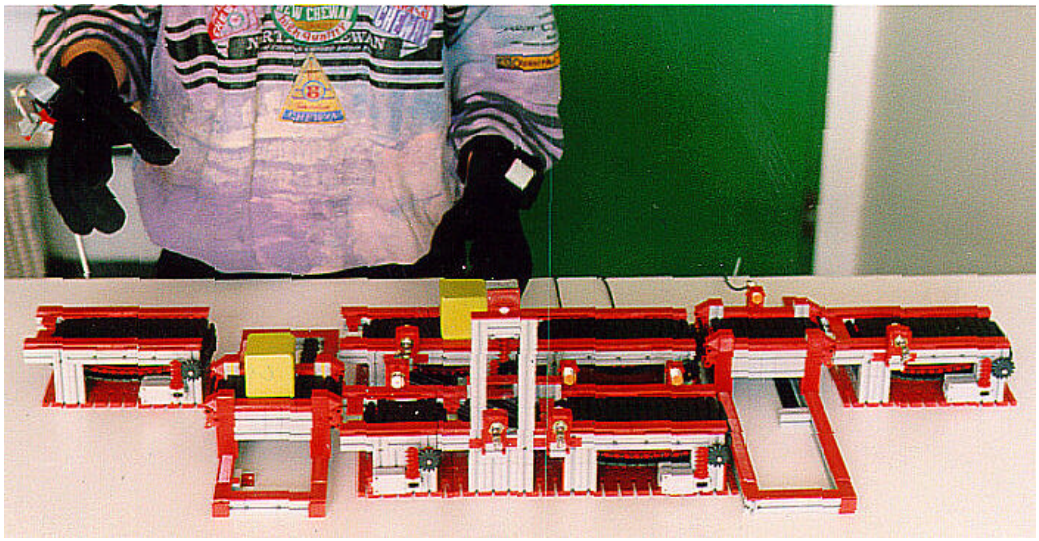

Bild 16.: Detailliertes Modell einer Bearbeitungszelle des EUROSIM-Szenarios

Der virtuelle Teil der Twin-Objects dient der Repräsentation des Modells im Computer. Sie erfüllen vier Aufgaben:

1. Sie repräsentieren und Speichern die Geometrie und Topologie des virtuellen Modells. Hierfür werden ihnen die Attribute Position und Orientierung zugeordnet.
2. Für die Grifferkennung werden die Objekte von einer Bounding-Box umgeben. Hierfür werden die Längen-, Breiten- und Höhenmaße benötigt. Diese Anforderung wird durch die Zuordnung einer geometrischen Beschreibung erfüllt, aus der die notwendigen Maße ermittelt werden.
3. Zur Visualisierung der virtuellen Welt wird ebenfalls auf die Geometrie zurückgegriffen. Hierbei ist die Ähnlichkeit zu den Modellbausteinen von ausschlaggebender Bedeutung. Für die Visualisierung müssen die Flächen der Bausteine in Echtzeit dargestellt werden. Eine hohe Detailtreue mit vielen Polygonen führt zu einer verminderten Leistung (Wiederholrate der Bilder) des Systems. Bei der geometrischen Repräsentation muß deshalb ein Kompromiß zwischen Performance und Anschaulichkeit gefunden werden. Bild 21 (links) zeigt ein sehr detailliertes Förderbandmodell, bestehend aus ca. 2000 Polygonen. Dieses bietet eine hohe Anschaulichkeit, ist aber für eine Echtzeitdarstellung mit der vorhandenen Hardware nicht geeignet. Bild 21 (rechts) stellt einen Kompromiß dar. Eine Ähnlichkeit zu den realen Modellbausteinen ist vorhanden und eine geringe Anzahl von Polygonen erlaubt die schnelle grafische Darstellung auch bei komplexen Modellen.
4. Für die automatische Topologieanalyse und Simulation sind die Bausteine mit Sensepoints und dem Verweis auf eine Simulationsbausteinklasse, die ein Materialflußgrundverhalten für die Dynamik enthält, ausgestattet.

Die virtuellen Bausteine sind objekt-orientiert aufgebaut (vergl. SML-Format). Alle Objektinstanzen im Modell erben die Eigenschaften der Bausteinklasse. Auch Bausteine in fertig aufgebauten Szenen können durch Austausch der Klassen mit anderen Eigenschaften ausgestattet werden. Dadurch kann z.B. zur Modellierung eine andere grafische Repräsentation als für spätere Präsentationen verwendet werden. Für alle oben dargestellten Bausteine sind virtuelle Bausteinklassen vorhanden. Es stehen damit drei Sätze von Twin-Objects für verschiedene Zwecke zur Verfügung.

Bei der Arbeit mit den verschiedenen Bausteinkästen konnte beobachtet werden, daß sich der Detaillierungsgrad signifikant auf die Arbeitsweise auswirkt. Während abstrakte Bausteine, wie in Bild 9, ein schnelles, skizzenhaftes und abstraktes Modellieren fördern, implizieren detaillierte Bausteine wie z.B. in Bild 17 ff. ein genaueres, näher an realen Gegebenheiten orientiertes Modellieren. Der richtigen Auswahl eines Detaillierungsgrades kommt somit eine große Bedeutung zu. Aus unserer

Sicht sollte im Rahmen der fortschreitenden Planung eines Projekts der Detaillierungsgrad durch Austauschen der Bausteinsätze stufenweise erhöht werden. Die Top-Down Struktur, die sich hieraus ergibt, unterstützt somit Planungsprozesse, wie sie in der Industrie Praxis sind.

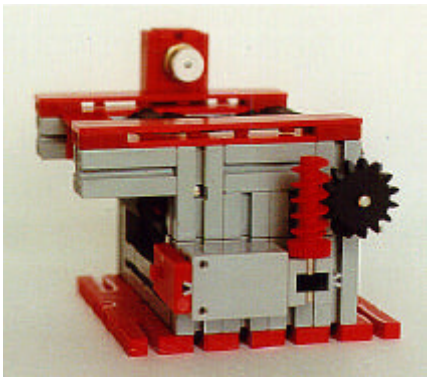

Bild 17.: Förderband 1 Längeneinheit

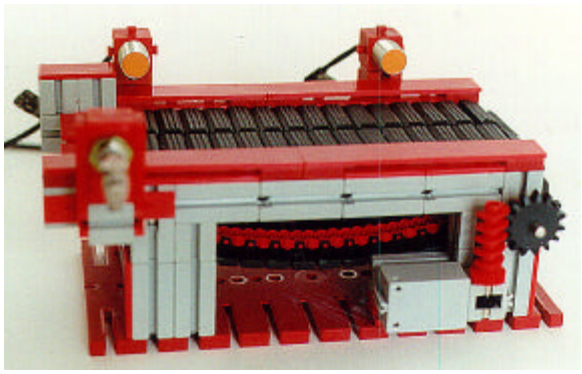

Bild 18.: Förderband 2 Längeneinheiten

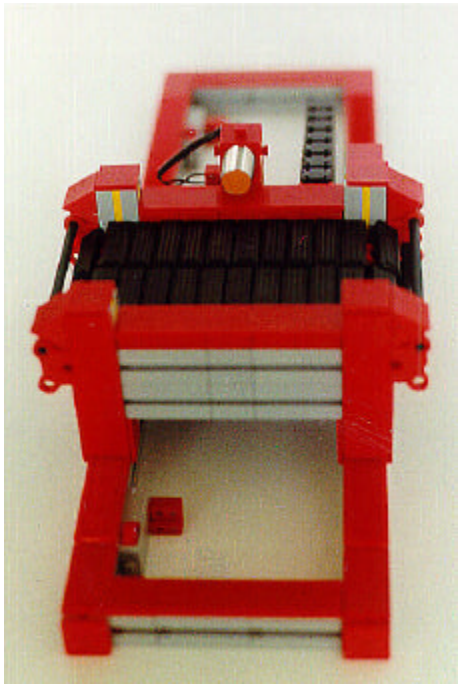

Bild 19.: Quertransport

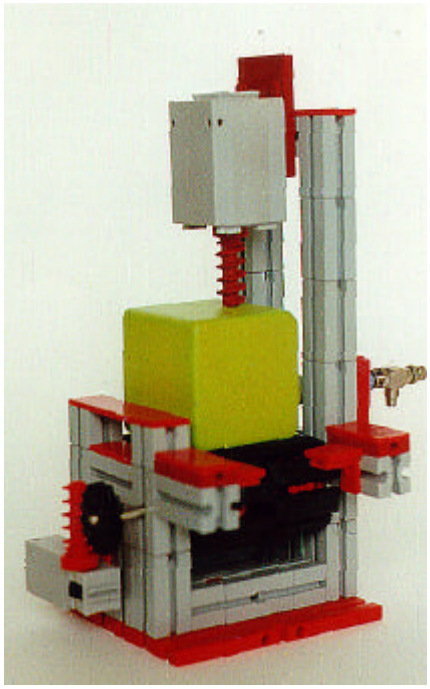

Bild 20.: Maschine

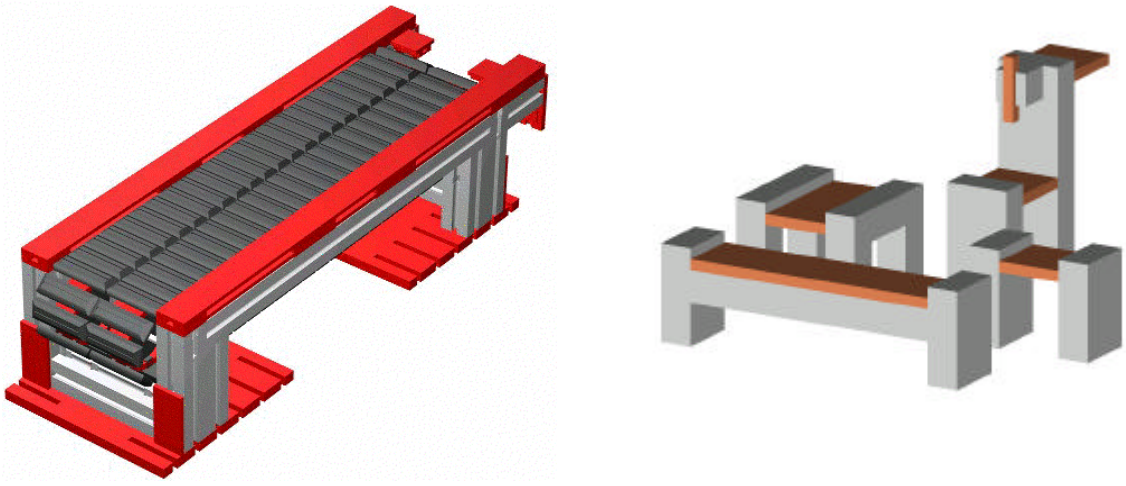

Bild 21: Virtuelle grafische Modelle von Bausteinen in verschieden Detaillierungsgraden.

1.6 Auswahl besonders geeigneter Anwendungsfälle

Mit dem bisher erreichten Stand der Entwicklung besteht eine gute Ausgangsbasis für die Ausweitung des Konzepts der Modellierung im Gegenständlichen auf weitere Anwendungsgebiete. Wir gehen davon aus, daß unser Ansatz, neben der im Projekt aufgegriffenen Problematik im Bereich der Simulationstechnik, auch auf allgemeinere Fragen der Modellierung zu beziehen ist. Es wurden eine Reihe von Themen ausgewählt, welche in verschiedenen Konstellationen behandelt werden:

- bezogen auf das laufende Projekt, soll die vorhandene Modellierungs-Umgebung zu einem Tele-Modellierer erweitert werden (siehe Antrag),
- die Anwendung unseres Konzeptes für die Konstruktion/CAD ist das Thema einer Diplomarbeit; aus elementaren Bausteinen (Fishertechnik, Lego) werden komplexe Teile zusammengesetzt,
- die Entwicklung eines VRML-Editors wird ebenfalls von einem Diplomanden betrieben. Aus gegenständlichen Bausteinen und aufgezeichneter Dynamik werden VRML-Welten konstruiert und im Internet publiziert,
- eine Stipendiatin und eine Diplomandin befassen sich mit der Frage, wie mit Hilfe deformierbarer Modelliermasse die Erstellung von Freiformflächen im Rechner unterstützt werden kann,
- ein weiteres DFG-Projekt untersucht die didaktischen Möglichkeiten der Gegenständlichkeit beim Aufbau von Pneumatik-Schaltungen im Berufsschulunterricht,

Zu den Firmen Festo Didactic, als bedeutendem Hersteller von realen produktionstechnschen Bausteinen, sowie Superscape Ltd. und Virtual Presence, als Hersteller und Entwickler von VR-Systemen und Bausteinen, wurden Kooperationsbeziehungen aufgebaut. Gemeinsame Projekte für die Entwicklung von Anwendungs- und Demonstrationssystemen befinden sich in der Planung.

2. Literatur

Aspern, J. (1993): SPS-Softwareentwicklung mit Petri-Netzen. Hüthing Verlag, Heidelberg.

Cypher, E. (Ed.) (1994): Watch What I Do - Programming by Demonstration. MIT Press, Cambridge, Massachusetts

Kämper, S. (1991): PEGROS. Ein Konzept zur Entwicklung eines graphischen objektorientierten Modellbildungs- und Simulationswerkzeugs auf der Basis von Petri-Netzen. Verlag Dr. Kovac, Hamburg.

Krauth, J. (1991): Comparison 2, Flexible Assembly System. EUROSIM Simulation News 1, p. 28, March 1991.

Krauth, J. (1992): Comparison 2, Flexible Assembly System. EUROSIM Simulation News 2, May 1992.

Krauth, J.; Meyer, R. (1995): Comparisons of Simulation Systems based on a Test Model. OPA, Amsterdam.

MacKenzie, C. L. & Iberall, T. (1994): The Grasping Hand. Elsevier Science Publishers, Amsterdam

Reisig, W. (1985): Systementwurf mit Netzen. Springer, Berlin...

Schäfer, K. (1995): Objektorientierte Modellierung zur Simulation des Steuerungsverhaltens von modularen Transfersystemen. Diplomarbeit, Universität Bremen

3. Projektbezogene Veröffentlichungen

Brauer, Volker (1996): Simulation Model Design in Physical Environments. ACM SIGGRAPH Computer Graphics, Vol. 30, No. 4, Nov. 1996.

Bruns, F. W., Schäfer, K. & Brauer, V. (1997): A new Approach to Human-Computer Interaction – Synchronous Modelling in Real and Virtual Spaces. Proceedings of Designing Interactive Systems (DIS 97), Amsterdam.

Schäfer, Kai (1997/98): Real Reality - Simulationsunterstützung durch gegenständliche Modelle. Tagungsband der Fachtagung Simulation und Visualisierung, Otto-von -Guernicke-Universität, Magdeburg.

Bruns, F.W. (1996): Grasping Communikating, Understanding - Connecting Reality and Virtuality. AI & Society, Oct. 96.

Bruns, F.W. (1997): Sinnlichkeit in der Technikgestaltung und Technikhandhabung - Ein konstruktiver Ansatz. In C. Schachtner (Hrsg.) Technik und Subjektivität. Suhrkamp, Frankfurt a. M., S 191-208.